EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

Instituto Politécnico Nacional
"La Técnica al Servicio de la Patria"

Research in Computing Science

ISSN: 1870-4069

Vol. 148 No. 12
December 2019

Research in Computing Science

Research in Computing Science, Año 18, Volumen 148, No. 12, diciembre de 2019, es una publicación mensual, editada por el Instituto Politécnico Nacional, a través del Centro de Investigación en Computación. Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, Ciudad de México, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor responsable: Dr. Grigori Sidorov. Reserva de Derechos al Uso Exclusivo del Título No. 04-2019-082310242100-203. ISSN: en trámite, ambos otorgados por el Instituto Politécnico Nacional de Derecho de Autor. Responsable de la última actualización de este número: el Centro de Investigación en Computación, Dr. Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Fecha de última modificación 01 de diciembre de 2020.

Research in Computing Science, year 18, Volume 148, No. 12, December 2019, is published monthly by the Center for Computing Research of IPN.

Volume 148(12)

Advances in Natural Language Processing

Alexander Gelbukh (ed.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2019

ISSN: 1870-4069

Formerly ISSN: 1665-9899.

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.rcs.cic.ipn.mx
http://www.ipn.mx
http://www.cic.ipn.mx

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

ISSN 1870-4069

Table of Contents

Page

ISSN 1870-4069

Learning Multi-Label Classification from Data Annotated with Unique Labels

Gargi Roy, Lipika Dey

Tata Consultancy Services Limited,
Research and Innovation, Kolkata,
India

{roy.gargi, lipika.dey}@tcs.com

Abstract. Classifying consumer-generated text like feedbacks, surveys or support emails has gained tremendous popularity in recent years as these contain important information about customer problems, opinions or processes. Given the volumes and velocity at which such text is generated, manual analysis is impossible. The most common analytical technique applied is classification of these texts into known categories, determined based on domain expertise. However, each piece of text may contain many different issues which even human agents fail to detect correctly. They usually classify the text based on the most important issue or the one occurring first while ignoring the others. The most suitable method for automatically analyzing these texts is to use multi-label classification. However, the training data available usually has single labels associated with them. This work proposes a classifier that learns from data annotated with single labels but predicts multi-label outputs along with confidence values. The proposed method is evaluated on different types of data sets, including those with noisy texts. Multi-labeled output provides richer insights.

Keywords: Classification, multi-label, unique annotation, supervised term weighting.

1 Introduction

As digitization grows rapidly, there is an increasing demand for intelligent text classification to automatically handle large volumes of consumer-generated text in the form of emails, customer complaint logs, call-logs, product reviews on social media and so on. Despite the existence of a multitude of techniques, classification of these kinds of noisy texts remains a challenging task. Real world documents like these need to be assigned multiple labels corresponding to the multiple categories of content contained in them, which most of the existing techniques cannot do. A major challenge is due to the lack of properly annotated multi-label training data. Class boundaries are also ill-defined and overlapping in real data. Manual annotation is often noisy and incomplete in a sense that a document which actually belongs to multiple classes usually gets single class annotation which corresponds to either the most obvious issue or the first issue that is mentioned in the text [6].

This is elaborated with the following customer complaint log from a mobile operator. *"Dear Sir I m buy a new (mobile phone name) on (date). on the box (service provider (SP) name) free data offer and i used already a (SP name) GSM sim (sim number) and i use this sim in the (phone name) but data offer 500mb/month for 6 month not activate on my (SP name) no. and i call (SP name) customer care they don't answer my problem. and i go to nearest (SP name) store they not listen my problem properly. so i m kindly request you plz solved my problem soon..."* [9]. This document is categorized by human agents as belonging to category "Internet Access Issue" and hence assigned that label.

However, ignoring the noise, it can be concluded that though this complaint primarily talks about Internet access issue related to the sim card, it also contains customer service related issues. If this document, and many similar to this one, are all assigned only to the single class of "Internet Access Issue", then the issue of bad customer service will never be highlighted. In fact, content pertaining to "customer service" may figure only in conjunction with a main functional business class, rather than by itself. Thus the problem we are trying to address can be defined as one which will learn from training instances uniquely classified as "Internet access issue", "sim related issue" or "customer service issue" etc. in the past - but assign more than one label correctly to future documents.

It may be noted that future documents may contain unique and varying combinations of class labels which have not been seen before. None of the existing multi-class or multi-label classifiers can handle learning from these types of incomplete labeling [3, 30, 33]. In this paper, we propose a classification method that learns to assign multiple class labels to unseen new documents even though it is trained with documents that had been assigned a single label. The proposed method thus corresponds to a scenario of learning from noisy or incompletely labeled training data. The algorithm uses supervised term weights learned from single-labeled (fixed set of labels) training samples.

The weights capture both the class-discriminating and class-representative powers of words. During classification, the relative distribution of different class labels within a given text is computed based on the strengths of each label assessed independently. The distribution is further analyzed to assign confidence measures to each label. The uniqueness of the proposed method lies in the fact that it can address the difficult task of learning fine-grained classification of content into multiple buckets from noisy training instances that are incompletely labeled. Our method is also explainable so that we can locate portions/words of the text corresponding to the different class labels along with their corresponding importance.

This is an important activity to handle consumer-generated text within a support environment, as described earlier in the example. Though deep learning based text classification techniques have gained wide popularity, the lack of explainability for these methods often make them unsuitable for adoption in such cases. For business applications traceability of a decision to its origin is mandatory in order to resolve possible future disputes. We have conducted extensive experiments and show that the proposed method outperforms several other classification algorithms for noisy data. The techniques can also handle data with rare classes and can be easily extended to

handle numeric or mixed data. We have compared our results with that of [20] which learns a fuzzy classifier for multi-label classification from single labeled mixed and numeric data. This paper is closest to our work but does not report any result for classifying unstructured data. The rest of the paper is organized as follows. Section 2 presents a review of related work. Section 3 presents the proposed term weighting mechanism. Section 4 presents the proposed classification technique. Section 5 presents results of experimentation with different data sets.

2 Literature Survey

Though many multi-label classification techniques have been proposed in the past, very few have considered the task of learning to assign multiple labels to unseen elements while training with uniquely labeled instances. None of the earlier work have reported results for text documents. A review of multi-label classification algorithms is presents in [33] which includes techniques that divides the problem into a set of binary classification problems [32] and multi-label ranking approach [11] considering pairwise relations between labels.

A fuzzy multi-label classification technique is presented in [10] for noisy data. In [20], a method is proposed to turn discriminative single-task classification into generative multi-task classification by adopting a fuzzy classification approach for numeric and mixed data. Authors in [29] also propose methods to provide crisp decisions for single labels and soft decisions for multi-labels while learning from single labeled numeric data. Methods for estimating confidence of binary classification output for a Case-Based Spam Filter is presented in [12].

3 Text Pre-Processing and Term Weighting

Text Preprocessing: Since consumer-generated digital text is fraught with noise, text cleaning is necessary. An array of noise removal modules proposed in [13] are implemented to improve the quality of text for classification purposes. These include handling text import errors such as presence of html code fragments, unicode characters, unnecessary duplication of characters or text elements that are found in social media.

A separate module is implemented to remove mandatory yet unwanted pieces text like signature or disclaimers within emails, advertisement or anchor text in web pages etc. The preprocessing module also removes unconventional use of punctuation marks, symbols and emoticons. Finally, a domain-dependent spell corrector, also explained in [13], is employed to automatically correct misspelt words.

This module first learns the frequently occurring non-dictionary words of a domain from the training set using the statistical distribution of the words and then corrects the misspelt words based on their similarity to the extended dictionary. This is an important step since mistakes or variations in words can drastically affect the word weights and in turn affect the performance of the classifier.

Stop-words are removed and all words are stemmed using [25] before computing the weights. **Term Weighting:** In this work we have used a term weighting scheme that considers two aspects of a word - (i). its class-discriminating power and (ii). its

class-representative power. The class-discriminating power of a word is measured using Inverse Gravity Moment (IGM) introduced in [7]. Equation 1 presents the computation of IGM, $f_{k_1} \geq f_{k_2} \geq ... \geq f_{k_p}$ denote the total number of occurrence of the term t_k in each of the classes sorted in descending order with p number of class labels, λ is an adjustable coefficient.

$$\text{IGM} = 1 + \lambda \cdot \left(\frac{f_{k_1}}{\sum_{r=1}^{p} f_{k_r} \cdot r} \right). \tag{1}$$

It can be seen that the higher the non-uniformity of distribution of a term across classes in the whole corpus, the greater is its class distinguishing power. The class-representative power is captured as a function of intra-class frequency. We have worked with following three different weighing mechanisms (denoted by $w^d_{\text{(L/N/R)TF-IGM}}(t_k)$) for term t_k, where t^d_k denote the number of occurrence of term t_k in document d. *i*) NTF: which uses the term frequency i.e. the raw term count normalized by document length.

$$w^d_{\text{NTF-IGM}}(t_k) = \left(\frac{t^d_k}{\#\text{terms in } d} \right) \left(1 + \lambda \cdot \frac{f_{k_1}}{\sum_{r=1}^{p} f_{kr} \cdot r} \right). \tag{2}$$

ii) LTF: uses the logarithm of raw term count, thus introducing a damping factor that does not allow the weight of a word to grow linearly with its frequency in a class.

$$w^d_{\text{LTF-IGM}}(t_k) = \log(t^d_k + 1) \left(1 + \lambda \cdot \frac{f_{k_1}}{\sum_{r=1}^{p} f_{kr} \cdot r} \right). \tag{3}$$

iii) RTF: uses the square root of raw term count which is another damping function.

$$w^d_{\text{RTF-IGM}}(t_k) = \sqrt{t^d_k} \left(1 + \lambda \cdot \frac{f_{k_1}}{\sum_{r=1}^{p} f_{kr} \cdot r} \right). \tag{4}$$

IGM is calculated once during parsing the training corpus and saved.

4 Proposed Classification Method

Given a set of documents D where each document is associated with a unique label from a set of class labels C, a term vocabulary T of size n and a term-document weight computation mechanism as stated earlier, we now present how multi-label class associations are generated for a new document d.

Let us assume that there are p number of class labels, $C = \{c_1, c_2, ..., c_p\}$. Let D_i denote the subset of documents in D that have the single class-label c_i associated to them. Let $\{o_1(d), o_2(d), ...o_p(d)\}$ denote the multi-class membership values that are initially computed for d for all classes, based on the words contained in it. Computation of $o_i(d)$ is computed as a function of the term weights for each class label *i*, which in turn is computed as an aggregate of the term-document weights for all training instances labeled with *i*:

$$o_i(d) = \sum_{j=1}^{m} \widehat{z}^{D_i}(t_j) w^d_{\text{(N/L/R)TF-IGM}}(t_j). \tag{5}$$

where m denotes the number of terms from T contained in d, $\text{IGM}^D(t_j)$ is the IGM value computed for term t_j in the document set D, $w^d_{\text{(N/L/R)TF-IGM}}(t_j)$ is computed as follows:

$$w^d_{\text{(N/L/R)TF-IGM}}(t_j) = w^d_{\text{(N/L/R)TF}}(t_j) * \text{IGM}^D(t_j). \tag{6}$$

$\widehat{z}^{D_i}(t_j)$ denotes the relative significance of term t_j for class c_i against all classes, computed from class-association of documents in D_i:

$$\widehat{z}^{D_i}(t_j) = \widehat{y}^{D_i}(t_j) / \sum_{r=1}^{(p)} \widehat{y}^{D_r}(t_j). \tag{7}$$

where $\widehat{y}^D_i(t_j)$ is computed as a normalized weight with respect to the vocabulary T as shown below where, $v^{D_i}(t_j)$ is the weight computed for a term t_j using one of the term weighting schemes presented in the section 3:

$$\widehat{y}^{D_i}(t_j) = v^{D_i}(t_j) / \sum_{m=1}^{n} v^{D_i}(t_m). \tag{8}$$

For an imbalanced data set, where the class distribution for the instances is very skewed, an additional normalization is done as follows:

$$\widehat{y}^{D_i}(t_j) = \widehat{y}^{D_i}(t_j) / \text{max}(\widehat{y}^{D_i}(t_{j=1 \text{ to } n})). \tag{9}$$

Algorithm 1 presents the computational steps. The class membership values obtained for d, are further normalized as shown in Step 7 of the algorithm and the class membership distribution is stored as a vector $\mathbf{X}$. Algorithm 1 has complexity $O(pm^2)$.

ALGORITHM 1: ComputeClassMembershipDistribution(D, T, d)

Input : D, T, d
Output: $\{o_1, o_2, ...o_p\}$ for d

Compute $\mathbf{Y_{p\times n}}$ from D, $\widehat{y}_{ij} \leftarrow v_{ij} / \sum_{j=1}^{n} v_{ij}$ where:
$v_{ij} \leftarrow \Sigma_{\text{document } d' \in D \text{ has label } i}(w(t_j^{d'}))$;
if *if imbalanced data* **then**
  $\widehat{y}_{ij} = \widehat{y}_{ij} / \max(\widehat{y}_{ij=1 \text{ to } n})$
Compute $\mathbf{Z_{p\times n}}$ from D where $\widehat{z}_{ij} = \widehat{y}_{ij} / \sum_{i=1}^{p} \widehat{y}_{ij}$;
Calculate term weight vector $\mathbf{N_{1\times m}}$ from d;
for d compute membership value for each class in matrix $\mathbf{O_{p\times 1}}$ where $\mathbf{O}_{p\times 1} = \mathbf{Z}_{p\times m}\mathbf{N^T}_{m\times 1}$;
Normalize class membership values, $o_i(d)^{\text{updated}} = o_i(d) / \sum_{i=1}^{p} o_i(d)$

Final Prediction with Confidence Category and Score: The distribution vector $\mathbf{X}$ which contains the strength of each class in the document is analyzed to generate the final predicted label set along with a confidence category and score associated with the predicted label set. We now define few terms.

Given p number of class labels and the class membership distribution for a new instance as $\mathbf{X} = \{x_1, x_2, ..., x_p\}$, let $x_\mu, \bar{x}, \sigma^2, \gamma, \kappa$ denote the maximum value, mean, variance, skewness and kurtosis of the distribution $\mathbf{X}$ respectively. Let $\dot{x_i}$ denote the percentage deviation of $x_i \in \mathbf{X}$ from mean of $\mathbf{X}$, which is:

$$\dot{x_i} = \frac{x_i - \bar{x}}{x_i} \cdot 100. \tag{10}$$

We then compute a score $\psi(x_i)$ for each $x_i \in \mathbf{X}$ as follows which takes into account the relative label strengths of the predicted labels with respect to the mean along with the number of labels, variability and degree of symmetry of the distribution and combined weight of the tails relative to the rest of the distribution to give insight about the significant peak in a distribution considering several structural characteristics of the distribution:

$$\psi(x_i) = \frac{\dot{x_i}}{100} \cdot p \cdot \sigma^2 \cdot |\gamma + \kappa|. \tag{11}$$

Let $\mathbf{X}_i^\eta$ denote the set of values within $\eta\%$ boundary of x_i in the distribution $\mathbf{X}$ derived as follows:

$$\mathbf{X}_i^\eta = \{y_i\} \text{ such that } y_i \in \mathbf{X} \text{ and } y_i \in \left(x_i, \ \frac{x_i \cdot (100 - \eta)}{100}\right). \tag{12}$$

Let α and β be two parameters which denote the upper and lowers boundaries in determining different confidence categories. α, β and η are parameters that control the degree of flexibility or rigidity allowed while interpreting the results. Now we present the determination of the final output label set along with the confidence categories using the above mentioned terms.

The first four categorization is done for $\left|\sum_{i=1}^{p}\psi(x_i)\right| > (0+\rho), x_i \in \mathbf{X}$, where ρ is an adjustable parameter.

i) Single Label with Very High confidence (SLVH): When the class-membership distribution has a distinct peak which is significantly higher than the rest satisfying the following conditions, then the class-label corresponding to the peak is inferred with very high confidence:

$$((\dot{x}_\mu > \alpha) \wedge (\psi(\dot{x}_\mu) > 0)) \wedge (((|\mathbf{X}_\mu^\eta| = 0) \vee ((|\mathbf{X}_\mu^\eta| > 1) \wedge ((\nexists x_i \in \mathbf{X}_\mu^\eta) \wedge (\dot{x}_i > \alpha))))). \quad (13)$$

ii) Multi-Label with High confidence (MLH): The class distribution has multiple high peaks satisfying the following conditions, those high-valued labels are predicted with high confidence:

$$((\dot{x}_\mu > \alpha) \wedge (\psi(\dot{x}_\mu) > 0) \wedge (|\mathbf{X}_\mu^\eta| > 1)) \wedge (\exists x_i \in \mathbf{X}_\mu^\eta \wedge (\dot{x}_i > \alpha) \wedge (\psi(\dot{x}_i) > 0)). \quad (14)$$

iii) Single Label with Medium confidence (SLM): The distribution has a maximum peak (satisfying the following conditions), which is not significantly higher than the mean, then the peak-valued label is predicted with medium confidence:

$$((\alpha \geq \dot{x}_\mu > \beta) \wedge (\psi(\dot{x}_\mu) > 0)) \wedge (((|\mathbf{X}_\mu^\eta| = 0) \vee ((|\mathbf{X}_\mu^\eta| > 1) \wedge (\nexists x_i \in \mathbf{X}_\mu^\eta \wedge (\alpha \geq \dot{x}_i > \beta))))). \quad (15)$$

iv) Multi-Label with Medium confidence (MLM): The distribution contains more than one peak, satisfying the following conditions, though not one with very high amplitude, multiple labels corresponding to the peaks are output, with medium confidence:

$$((\alpha \geq \dot{x}_\mu > \beta) \wedge (\psi(\dot{x}_\mu) > 0) \wedge (|\mathbf{X}_\mu^\eta| > 1)) \wedge (\exists x_i \in \mathbf{X}_\mu^\eta \wedge (\alpha \geq \dot{x}_i > \beta) \wedge (\psi(\dot{x}_i) > 0)).. \quad (16)$$

v) Reject Classification for LOW confidence (RCLC): The distribution is flat type with almost uniform low values for each label, the prediction result is not accepted due to low confidence. The condition is as follows:

$$(\beta \geq \dot{x}_\mu) \wedge \left(\left|\sum_{i=1}^{p}\psi(x_i)\right| \approx 0, \forall x_i \in \mathbf{X}\right). \quad (17)$$

The final label set contains the predicted labels in order such that the first label is the most significant one and so on. After the final label set determination, the confidence score is computed in the following manner. Finally, the scores are normalized:

$$s = (\text{Avg}(\psi(x_i), \forall x_i \in \text{output label set}) * \left(\left|\sum_{i=1}^{p}\psi(x_i)\right|, \forall x_i \in \mathbf{X}\right). \quad (18)$$

5 Results and Comparative Study

This section presents the dataset description, classifier results along with comparative results with another relevant work. **Evaluation scheme:** The evaluation of this work can not be done in the straight forward manner. We have evaluated our results from different aspects. As no baseline is available for this kind of work for unstructured data,

Table 1. Performance measures of 20 Newsgroup data from other reported works and this paper.

Results reported by	Classifier	Macro F1	Accuracy
[17]	Naive Bayes	83	
	Rocchio	78.6	
	K-NN	81.2	
	SVM	86	
[2]	SVM	78.19	
	L-Square	83.05	
[24]	SVM (CS&T)	82.4	
	LR (CS&T)	81.5	
[16]	RSV-NN	83	
[15]	GE1-MNB	63	
	MaxEnt	79	
[8]	LSTM		82
	LM-LSTM		84.7
	SA-LSTM		84.4
[31]	SC-LSTM-P		82.98
[4]	CNN2		80.19
This paper	**-**	**84.7**	**84.87**

we first compute the standard performance measures such as precision, recall, macro F1 score, accuracy using the first label which is also the most significant label in the output label set in ten fold cross validation setup. Then using the multi-label output set, the overall prediction performance is updated by updating the confusion matrix as when given annotation does not match with the first label with highest membership value but matches with the second or third highest in the multi-label output. We have compared our results with the other reported works including deep learning techniques for 20 Newsgroup data.

We also have analyzed the labels from multi-label output set to see if they have some similarities among them which depicts the relevance of the output with respect to the given text. Then we analyse the interpretation of the predicted multi-labels by inspecting which words or portion of text have contributed to which predicted label with how much weight to verify the results. Our method is extended for structured data and results are compared with an existing baseline for structured data.

Dataset Description: Ten uniquely labeled datasets are considered for the experimentation. Among them eight are text datasets covering noisy, class overlapping, class imbalanced, short text, long text datasets and two are numeric and mixed datasets, Autos and Zoo, retrieved from UCI repository [19]. Text datasets are as follows, where points 5 and 6 have non-uniform class distribution with rare classes and multi-label aspect in significant amount.

1. Full 20 Newsgroups corpus data [18] having around 20,000 instances with 20 classes.

2. 'DisjointNews': 5000 instances from five top level disjoint classes within 20 Newsgroups corpus.

3. 'ScienceNews': 4000 instances within the science class of the 20 Newsgroups corpus.

4. 'CompScNews': 5000 instances within the computer class of the 20 Newsgroups corpus having class overlaps.

5. 'HR Internal': internal enterprise email communications corresponding to several enterprise activities (16,356 instances with seven classes, due to confidentiality issue class names are made anonymous).

6. 'Telecom': telecom customer complaint data from a publicly accessible consumer complaint management website [14] used in [10] (500 instances and six classes).

7. 'IMDB': IMDB movie review sentiment dataset, proposed by [21] [1] (25,000 instances with two classes).

8. 'RT': Rotten Tomatoes dataset [23][2] (10.662 instances, short text like one line, with two classes).

Prediction: Ten fold cross validation is done on the datasets to generate average standard performance measures. Results are generated using four different term weighing schemes such as considering only local factor NTF and then considering both local and global factor LTF-IGM, NTF-IGM, RTF-IGM using $\lambda = 7$ (as empirically found and used in [7]).

Table 1 presents some state-of-the-art classification performance including deep learning techniques published for the full 20 Newsgroups dataset along with the proposed classifier, which shows that our result is comparable with the existing methods. As no baseline is available for our work, we initially present the performance measures using the first label in the predicted label set which is presented in Table 2 for all the text datasets.

Prediction Enhancement with Confidence: Table 3 shows the enhanced classification performance (using the multiple labels in the predicted output) with confidence for CompScNews, HR (internal), Telecom and full 20 Newsgroups datasets. The results depict that there are instances which actually belong to multiple labels with varying confidence and for many instances first predicted label does not match with the given annotation, however, the multi-label output contains the true class, this, in essence enhance the classification performance.

Also there are many instances for which the module rejects the predicted result due to low confidence and is seen that predicted single label is incorrect for many of those instances. Although we found that the 20 Newsgroup data is not fully single-labeled which is also supported by [26, 28, 1], however, we have found that around 4% of the instances show multi-label aspect. It is seen that many classes with some overlaps have appeared in the multi-label output (shown in Figure 1), such as religion.misc and politics.guns (also found out by [27]), religion.misc and atheism, christian and religion.misc, politics.misc and politics.guns, pc.hardware and windows.misc etc.

[1] http://ai.stanford.edu/ amaas/data/sentiment/

[2] http://www.cs.cornell.edu/people/pabo/movie-review-data/

Table 2. Prediction performances using first label for text datasets. Model training for imbalanced data is used for HR (internal) data.

Dataset	Performance measures	Term weighting scheme			
		NTF	LTF-IGM	NTF-IGM	RTF-IGM
20 News (Full)	Macro F1	82.99	84.10	84.10	84.2
	Accuracy	83.53	84.49	84.49	**84.56**
ScienceNews	Macro F1	95.59	96.82	96.67	96.79
	Accuracy	95.6	**96.82**	96.67	96.8
DisjointNews	Macro F1	97.35	98.29	98.32	98.28
	Accuracy	97.35	98.3	**98.32**	98.28
CompScNews	Macro F1	83.46	85.73	86.27	85.68
	Accuracy	83.58	85.8	86.32	85.74
HR (internal)	Macro F1	80.37	85.15	84.47	85.08
	Accuracy	82.21	84.86	84.89	84.56
Telecom	Macro F1	60.35	61.02	63.1	60.4
	Accuracy	69.6	64	70.4	69
IMDB	Macro F1	86.06	87.63	87.89	87.62
	Accuracy	86.07	87.64	87.9	87.62
RT	Macro F1	75.34	79.03	78.85	79.05
	Accuracy	75.35	79.04	78.87	**79.06**

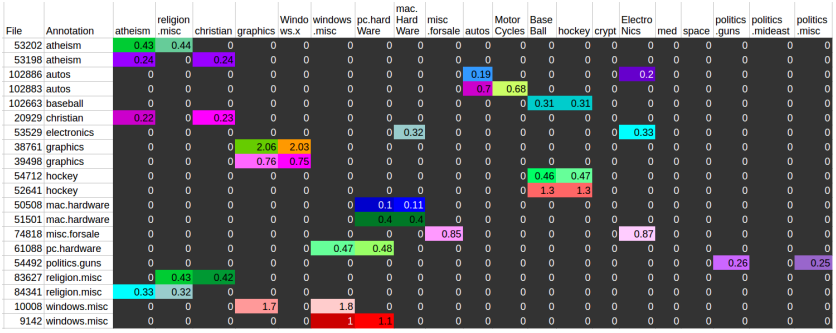


Fig. 1. Few instances with multi-label output from 20 Newsgroups dataset.

After final prediction and generating confidence scores, interpretability of the multi-label output is done by tracing the contributory words for each class. Figure 2 shows this result interpretability for the customer complaint presented in the section 1 from Telecom dataset.

Comparative Study with Fuzzy Classifier: [20] proposes an approach by turning discriminative single-task classification into generative multi-task classification by adopting a fuzzy rule induction approach implemented on the KNIME platform [5], empirically they have shown that an instance can belong to multiple classes in spite of its single annotation. Both the fuzzy rule induction technique of KNIME and and our classifier are run on the two UCI datasets, autos and zoo, to compare the class membership distribution.

Table 3. Statistics for several confidence categories along with updated performance. Freq.:frequency, FirstLabel: First Label of the predicted output matches with the given annotation, MultiLabel: Given annotation does not match with first Label but matches with the second or third label in the multi-label output set, FirstLabel′: First Label of the predicted output does not matches with the given annotation.

Category	Statistics	Dataset (% of total)			
		CompScNews	HR (internal)	Telecom	20 News
SLVH	Frequency	1405 (28%)	8372 (51%)	214 (42%)	18876 (94%)
	FirstLabel	1377 (27%)	6087 (37%)	189 (37%)	16432 (82%)
MLH	Frequency	0	17	0	930 (4%)
	FirstLabel	0	5	0	390 (1%)
	MultiLabel	0	5	0	367 (1%)
SLM, MLM	SLM Freq	3436 (68%)	6842 (41%)	240 (48%)	43
	MLM Freq	16	757 (4%)	33 (6%)	106
	FirstLabel	2879 (57%)	3989 (24%)	156 (31%)	37
	MultiLabel	9	330 (2%)	15 (3%)	49
RCLC	Frequency	130 (2%)	91	13 (2%)	0
	FirstLabel	49	19	7 (1%)	0
	FirstLabel’	81 (1%)	72	6 (1%)	0
	Macro F1	86.3	85.27	64.1	84.7
	Accuracy	**86.35**	**85.61**	**71.26**	**84.87**

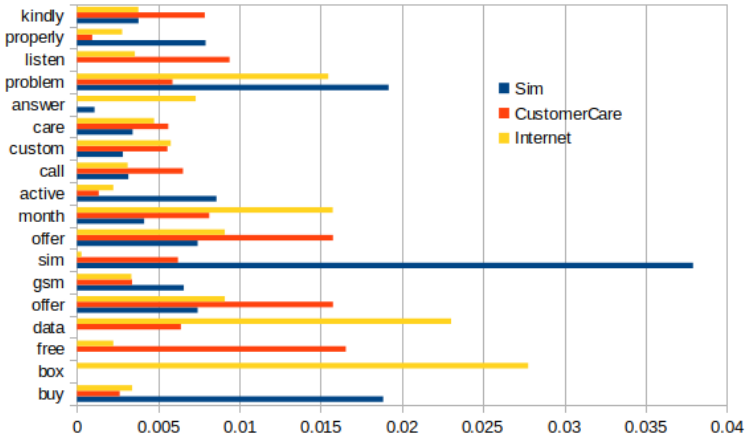


Fig. 2. Interpretability of the predicted label set, {Sim, Internet, CustomerCare} for the customer complaint presented in the section 1 from Telecom dataset.

Before applying our classifier, different kinds of data preprocessing techniques such as data scaling, data standardization, binarization of the categorical features, missing value handling through imputation are done. Mutual information [22] is used for feature selection.The results for autos dataset (representative instances) is given in the Table 4 and for most of the instances it is seen that their are similarities in the two distributions although our distributions range within [0,1] and the other is fuzzy degree.

Table 4. Few representative sample results are presented which are obtained from running KNIME and our technique on the autos dataset. NC: Normalized confidence.

Classifier	ID	Annotation	Class name (Dataset: Autos)					Output	Confidence category	NC
			-1	0	1	2	3			
KNIME	3	2	0.36	1	0.18	0.5	1	0	-	-
OUR			0.237	0.236	0.195	0.199	0.13	-1	SLM	0.21
KNIME	15	0	0.71	1	0	0.33	0.63	0	-	-
OUR			0.24	0.243	0.18	0.19	0.14	0	SLM	0.24
KNIME	66	0	0.25	0.81	0.076	0.61	0	0	-	-
OUR			0.239	0.245	0.187	0.2	0.12	0	SLM	0.09
KNIME	67	-1	0.37	1	0	0	0	0	-	-
OUR			0.244	0.249	0.17	0.2	0.13	0	SLM	0.37
KNIME	123	-1	0.5	1	0	0	0	0	-	-
OUR			0.23	0.26	0.21	0.18	0.11	0	SLM	0.07
KNIME	183	2	0	0	0.54	0.71	0	0	-	-
OUR			0.19	0.19	0.21	0.23	0.17	2	SLM	0.08
KNIME	192	0	0.26	0.45	0	0.54	0	2	-	-
OUR			0.248	0.242	0.18	0.21	0.12	-1	SLM	0.19
KNIME	202	-1	0.46	0.2	0.17	0.5	0	2	-	-
OUR			0.32	0.23	0.16	0.17	0.12	-1	SLM	0.73

Table 5. Few representative sample results are presented which are obtained from running KNIME and our technique on the zoo dataset. NC: Normalized confidence.

Classifier	ID	Annotation	Class name (Dataset: Zoo)							Output	Confidence category	NC
			1	2	3	4	5	6	7			
KNIME	5	1	**1**	0	0	0	0	0	0	1	-	-
OUR			**0.42**	0.1	0.13	0.12	0.12	0.09	0.02	1	SLM	0.83
KNIME	13	7	0	0	**1**	0	0	0	0	3	-	-
OUR			0.07	0.14	0.17	0.16	0.18	0.1	**0.19**	7	SLM	0.07
KNIME	21	2	0	**1**	0	0	0	0	0	2	-	-
OUR			0.09	**0.33**	0.11	0.13	0.14	0.13	0.07	2	SLM	0.56
KNIME	25	5	0	0	0	0	**1**	0	0	5	-	-
OUR			0.14	0.13	0.16	0.19	**0.21**	0.07	0.11	5	SLM	0.06
KNIME	30	6	0	0	0	0	0	**1**	0	6	-	-
OUR			0.07	0.27	0.1	0.06	0.12	**0.29**	0.08	6	SLM	0.07
KNIME	34	4	0	0	0.54	0.71	0	0	**1**	4	-	-
OUR			0.14	0.14	0.17	**0.23**	**0.21**	0.03	0.08	4, 5	SLM	0.12
KNIME	60	4	0	0	0	**1**	0	0	0	3	-	-
OUR			0.14	0.13	0.16	**0.23**	**0.21**	0.03	0.1	4, 5	SLM	0.08
KNIME	73	4	0	0	0	**1**	0	0	0	4	-	-
OUR			0.14	0.14	0.16	**0.24**	**0.21**	0.03	0.08	4, 5	SLM	0.12

For the zoo dataset, shown in Table 5, the fuzzy classifier, apart from correct single class prediction, fails to capture any multi-label aspect, however, interestingly our technique, along with correct prediction, captured a weak label association between two classes 4 and 5. Where class 4 and 5 consist of instances such as {catfish, piranha, seahorse, tuna} and {frog, toad, newt} respectively.

Instances of class 4 belongs to aquatic and class 5 belongs to both aquatic and terrestrial and both the classes are predator and tooted. The results indicate that an instance can belong to multiple classes in spite of its single annotation.

6 Conclusion

With the emergence of large volume of consumer generated data, demand for interpretable methods to mine business insights from those data also emerges. In many cases, each piece of text contains multiple issues which requires multi-label classification. However, most of the available data have single label annotation as manual annotation is costly and sometimes incomplete. Hence, this work proposes a classifier that learns from data annotated with single labels but predicts multi-label outputs. Thereafter a class-confidence computation mechanism is also proposed.

References

1. Ahmed, M. S., Khan, L., Oza, N. C., Rajeswari, M.: Multi-label asrs dataset classification using semi supervised subspace clustering. In: CIDU. pp. 285–299 (2010)
2. Al Mubaid, H., Umair, S. A.: A new text categorization technique using distributional clustering and learning logic. IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 9, pp. 1156–1165 (2006)
3. Aly, M.: Survey on multiclass classification methods. Neural Netw, vol. 19 (2005)
4. Arras, L., Horn, F., Montavon, G., Müller, K.-R., Samek, W.: What is relevant in a text document?: An interpretable machine learning approach. PloS one, vol. 12, no. 8, pp. e0181142 (2017)
5. Berthold, M. R., Wiswedel, B., Gabriel, T. R.: Fuzzy logic in knime–modules for approximate reasoning–. International Journal of Computational Intelligence Systems, vol. 6, no. sup1, pp. 34–45 (2013)
6. Bucak, S. S., Jin, R., Jain, A. K.: Multi-label learning with incomplete class assignments. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on. pp. 2801–2808. IEEE (2011)
7. Chen, K., Zhang, Z., Long, J., Zhang, H.: Turning from TF-IDF to TF-IGM for term weighting in text classification. Expert Systems with Applications, vol. 66, pp. 245–260 (2016)
8. Dai, A. M., Le, Q. V.: Semi-supervised sequence learning. In: Advances in neural information processing systems. pp. 3079–3087 (2015)
9. Dasgupta, T., Dey, L.: Multi-label classification and analysis of customer complaint logs. In: Workshop on Enterprise Intelligence, Knowledge and Data Discovery (KDD). ACM (2016)
10. Dasgupta, T., Dey, L., Verma, I.: Fuzzy multi-label classification of customer complaint logs under noisy environment. In: International Joint Conference on Rough Sets. pp. 376–385. Springer (2016)
11. Dekel, O., Singer, Y., Manning, C. D.: Log-linear models for label ranking. In: Advances in neural information processing systems. pp. 497–504 (2004)
12. Delany, S. J., Cunningham, P., Doyle, D., Zamolotskikh, A.: Generating estimates of classification confidence for a case-based spam filter. In: ICCBR. vol. 3620, pp. 177–190. Springer (2005)

13. Dey, L., Roy, G.: Auto-correction of consumer generated text in semi-formal environment. In: 7th Language and Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics. pp. 203–207. Fundacja Uniwersytetu im. Adama Mickiewicza w Poznaniu (November 2015)
14. Indian consumer complaint forum: Consumer complaints (2016)
15. Jin, Y., Wanvarie, D., Le, P.: Combining lightly-supervised text classification models for accurate contextual advertising. In: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers). vol. 1, pp. 545–554 (2017)
16. Ko, Y.: How to use negative class information for naive bayes classification. Information Processing & Management, vol. 53, no. 6, pp. 1255–1268 (2017)
17. Ko, Y., Park, J., Seo, J.: Improving text categorization using the importance of sentences. Information processing & management, vol. 40, no. 1, pp. 65–79 (2004)
18. Lang, K.: Newsweeder: Learning to filter netnews. Proceedings of the Twelfth International Conference on Machine Learning, pp. 331–339 (1995)
19. Lichman, M.: UCI machine learning repository (2013)
20. Liu, H., Cocea, M., Mohasseb, A., Bader, M.: Transformation of discriminative single-task classification into generative multi-task classification in machine learning context. In: Advanced Computational Intelligence (ICACI), 2017 Ninth International Conference on IEEE. pp. 66–73 (2017)
21. Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1. pp. 142–150. Association for Computational Linguistics (2011)
22. Murphy, K. P.: Machine learning: a probabilistic perspective. Cambridge, MA (2012)
23. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the 43rd annual meeting on association for computational linguistics. pp. 115–124. Association for Computational Linguistics (2005)
24. Parambath, S. P., Usunier, N., Grandvalet, Y.: Optimizing f-measures by cost-sensitive classification. In: Advances in Neural Information Processing Systems. pp. 2123–2131 (2014)
25. Porter, M. F.: An algorithm for suffix stripping. Program, vol. 14, no. 3, pp. 130–137 (1980)
26. Read, J.: Scalable multi-label classification. Ph.D. thesis, University of Waikato (2010)
27. Read, J., Bifet, A., Holmes, G., Pfahringer, B.: Efficient multi-label classification for evolving data streams, (2010)
28. Read, J., Perez Cruz, F., Bifet, A.: Deep learning in partially-labeled data streams. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing. pp. 954–959. ACM (2015)
29. Tang, W., Mao, K., Mak, L. O., Ng, G. W.: Classification for overlapping classes using optimized overlapping region detection and soft decision. In: Information Fusion (FUSION), 2010 13th Conference on. pp. 1–8 (2010)
30. Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. International Journal of Data Warehousing and Mining, vol. 3, no. 3 (2006)
31. Wang, Y., Tian, F.: Recurrent residual learning for sequence classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 938–943 (2016)
32. Zhang, M. L., Zhou, Z. H.: ML-KNN: A lazy learning approach to multi-label learning. Pattern recognition, vol. 40, no. 7, pp. 2038–2048 (2007)
33. Zhang, M. L., Zhou, Z. H.: A review on multi-label learning algorithms. IEEE transactions on knowledge and data engineering, vol. 26, no. 8, pp. 1819–1837 (2014)

ISSN 1870-4069

Investigating Deep Learning Approaches for Hate Speech Detection in Social Media

Prashant Kapil[1], Asif Ekbal[1], Dipankar Das[2]

[1] Department of Computer Science and Engineering,
Indian Institute of Technology Patna,
India

[2] Department of Computer Science and Engineering,
Jadavpur University Kolkata,
India

{prashant.pcs17, asif}@iitp.ac.in,
ddas@cse.jdvu.ac.in

Abstract. The phenomenal growth on the internet has helped in empowering individual's expressions, but the misuse of freedom of expression has also led to the increase of various cyber crimes and anti-social activities. Hate speech is one such issue that needs to be addressed very seriously as otherwise, this could pose threats to the integrity of the social fabrics. In this paper, we proposed deep learning approaches utilizing various embeddings for detecting various types of hate speeches in social media. Detecting hate speech from a large volume of text, especially tweets which contains limited contextual information also poses several practical challenges. Moreover, the varieties in user-generated data and the presence of various forms of hate speech makes it very challenging to identify the degree and intention of the message. Our experiments on three publicly available datasets of different domains shows a significant improvement in accuracy and F1-score.

Keywords: Hate speech, deep learning, F1-score.

1 Introduction

Social media is one platform that allows people across the globe to share their views and sentiments on various topics, but when it is intended to hurt some particular group or any individual then it is considered as hateful content. There is no such universally accepted definition of hate speech as it often varies across the different geographical regions. [28] stated that hate speech is an abusive speech with a high frequency of stereotypical words. It is demographic dependent as some countries allow some speech to be said under *Right to speech*, whereas other countries adhere to a very strict policy for the same message. In recent times, Germany made policy for the social media companies that they would have to face a penalty of 60$ million if they failed to remove illegal content on time. Denmark and Canada have laws that prohibit all the speeches that contain insulting or abusive content targeting minorities and could promote violence

and social disorders. The Indian government has also urged leading social media sites such as Facebook, Twitter to take necessary action against hate speech, especially those posts that hurt religious feelings and create social outrage. Setting aside legal actions our aim should be to combat these speeches by agreeing to a set of standard definitions, guidelines, and practices. [21] defined hate speech as any communication that demeans any person or any group based on race, color, gender, ethnicity, sexual orientation, and nationality.

Social networking sites like Twitter and Facebook are also taking preventive measures by deploying hundreds to thousands of staff to monitor and remove offensive content. [27] collected messages from Whisper and Twitter to define hate speech as any offense motivated, in whole or in a part, by the offender's bias against an aspect of a group of people. They investigated the main targets of hate speech in online social media and introduced new forms of hate that are not crimes but harmful.

The detection can't be done manually, rather it needs a thorough investigation of the techniques and build robust techniques to accomplish this task. The paper is structured as follows: We put the discussion on the related works in Section 2. Section 3 describes embeddings used, preprocessing and the model architecture. Datasets and experimental setup are described in Section 4. Results along with the error analysis to discuss the limitations of our proposed models are presented in Section 5. Finally, we conclude along with future work roadmaps in Section 6.

1.1 Motivation and Contribution

There has not been much research on hate speech detection because of the non-availability of annotated datasets as well as lack of proper attention to this field. Its detection is challenging as these are highly contextual and poses several challenges concerning the demographic characteristics and nature of the text.

The same message can be posted in different ways, with one could be the potential candidate for hate speech, while other is not. Data imbalance also introduces challenges to build a robust machine learning model. In this paper, we propose deep learning based approach to hate speech detection. We experimented with three publicly available benchmark datasets, i.e, [29, 9] and [15].

2 Related Work

Most of the previous works done in this area have used different data sets. Researchers have mostly used traditional machine learning algorithms, and recently have started using deep learning. Lexical based approaches misclassify any sentence containing slang indicative of hate, affecting *right to freedom of speech* as the word used may have different meaning used in some different contexts. [7] showed that support vector machine (SVM) with word-n-grams employed with syntactic and semantic information can achieve the best performance. [9] reported that using unigram, bigrams, and trigrams feature weighted with their TF-IDF values fed to logistic regression(LR) tends to perform best on their dataset by achieving 90% precision with hate class correctly predicted for 61% times. [26] classified ontological classes of harmful speech based on

the degree of content, intent, and affect that it is creating on social media. [29] used critical race theory to annotate a dataset of 16K tweets that is made publicly available. They observed that geographic and word length distributions do not have significant contributions in enhancing the performance of the classifier.

However, gender information combined with char-n-grams has shown a little improvement. [20] used four types of features like n-grams, linguistic features, syntactic features, and distributional semantic features to make a distinction between abusive and clean data in finance and news data. [2] made use of various semantic, sentiment and linguistic features to develop a cascaded ensemble learning classifier for identifying racist and radicalized intent on the Tumblr microblogging website.

[10] studied different forms of abusive behavior and made public the annotated corpus of 80K Tweets categorized into 8 labels. [30] classified 2010 sentences using features like unigrams, sentiment features, semantic features, and pattern-based features. [32] proposed a CNN-GRU based architecture that showed promising results for 6 out of 7 datasets, outperforming other state-of-the-art by 1-13 F1 points.

They also released a new dataset of 2435 tweets focusing on refugees and Muslims. [16] applied bag-of-words model to learn binary classifier for the labels *racist* and *non-racist* and achieved 76% accuracy. [5] used the combinations of neural network-based LSTM model with non-neural based GBDT representing words by random embedding and achieved the best result on the dataset of [29].

The method proposed in [22] focused on detecting abusive language first and then classify into specific types of abuse. They showed that hybrid CNN i.e a combination of char-cnn and word-cnn perform best over word-cnn and classical methods like logistic regression and svm on the dataset of 16K tweets by [29]. [11] showed the concept of using CNN with random vectors, word vectors based on semantic information, word vectors combined with character 4-grams, and compared the performance with each other.

3 Methodology

3.1 Pre-trained Word Vectors and One-Hot Encoding

a) W2V: We utilized the publicly available *word2vec* vectors trained on 100 billion words from Google news, trained using CBOW architecture [18] and have dimensions of 300.
b) GloVe [23]: Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. We used *glove.twitter.27B.100d* as the embeddings. For (a) and (b) All the out-of-vocabulary (OOV) words were assigned random weights in the range [-0.25, 0.25].
c) FastText: In our experiments, we also leveraged the skip-gram based approach by [6] that represents each word as a bag of character n-grams. A vector value is associated with each character, the sum of these vector values represent the embedding for words. The dimensions for these embedding are 300.
d) One-Hot encoding: The encoding is done by prescribing an alphabet of size m for the input language, and then quantize each character using 1-of-m encoding.

The alphabet used in all of our models consists of 27 characters, including 26 English letters and one for all other symbols.

3.2 Pre-processing

As the datasets have been crawled from social media, these contain noises and inconsistencies, such as slangs, misspelled words, acronyms, etc. Hence a light pre-processing is done by expanding all apostrophes containing words and then removing characters like : , & ! ?. The tokens were also converted to lower-case for normalization. We also used a dictionary to expand the misspelled words to its original form.

All the words starting with # were broken down into individual words using word segment in python. For e.g. *#KillerBlondes* becomes *killer blondes*, *#Feminism* becomes *feminism*, *#atblackface* becomes *at black face* and *#marriageequality* becomes *marriage equality* etc. Emoticons were also replaced with tokens like happy, sad, disgust, and anger.

3.3 Models

We developed 13 deep learning models using CNN, LSTM, BiLSTM, and Character-CNN. The models are described as follows. **CNN:** This model is based on the architecture by [14] that uses 5 main types of layers: *Input layer*, *Embedding Layer*, *Convolution layer*, *Pooling layer* and *Fully Connected layer.*

- Input Layer: All the sequences are converted to integer form where each token has been assigned a unique index. The input sequences are then zero-padded to have an equal length as it helps in improving performance by keeping information preserved at the borders.
- Embedding Layer: Each word w_i in the sequence is mapped to real-valued vector at the corresponding index in the embedding matrix using e(w_i), where *e* is the embedding matrix.
- Convolution Layer: It is used to extract features for better representation of data using the learnable filter of size i*h, where *i* is the window size and *h* is the embedding dimension. Each filter is convolved through *i* words at a time and performs an element-wise dot product to get a feature f_1. This process is repeated (n-h+1) times to get the feature map F = [f_1,f_2.....f_{n-h+1}]. *N* number of filters are used to get the different feature maps.
- Pooling Layer: It reduces the spatial size of the representation helping in reducing overfitting. Max pooling takes the local maximum value from the feature map depending on the pool size whereas global max pooling takes the pool size equal to the size of the input.
- Fully Connected layer: The vectorized form of features obtained from the last CNN layer is fed into the fully connected layer which has every input connected to every output by weight. This is followed by the softmax activation function that calculates the probability values for all the classes. Fig.1 describes the sample architecture of CNN with dimension = 5.

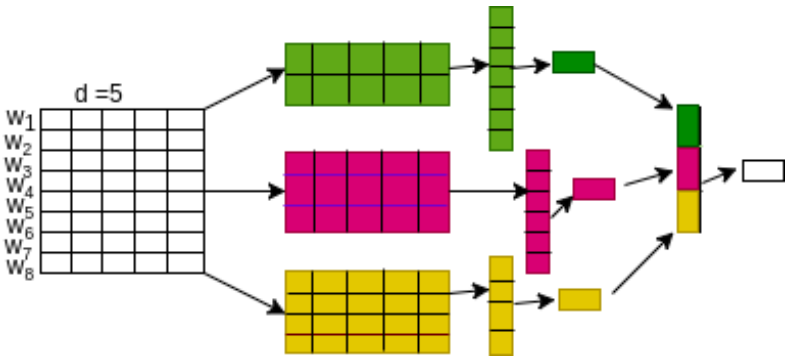


Fig. 1. Architecture of CNN.

Table 1. List of top occuring words in each class.

Class	High frequency words
Hate	b**ch, a**, nigger, f***ing, faggot, shit, trash, hate, kill, gay, ugly, queer, whitey
Offensive	b**ch, hoe, a**, ni**er, p***y, trash, wtf, crazy, stupid, p***s, gay, girl, hate
Racism	islam, religion, jews, women, war, christians, slave, terrorist, daesh, rape, beheaded
Sexism	sexist, women, girls, female, man, comedians, blondes, feminism, bitch, bimbos
Covertly	people, india, country, religious, party, political, muslims, fatwa, pakistan, modi, bjp
Overtly	people, india, religion, pakistan, bjp, muslims, hindu, terrorist, killed, fatwa

LSTM/BiLSTM: RNN is very suitable for sequence learning, time series but as it suffers from vanishing gradient and exploding gradient it does not perform well for the long-range dependency. So [13] introduced LSTM that is capable of learning long-range dependencies. The input sequence (i_1, i_2...i_n) is transformed into its vector form of embedding size e which is then converted to h_1=(h_1^1, h_2^1...h_n^1) and transferred to the successive layers. It works by learning only the past information of the sequence, however, Bi-LSTM i.e a variant of LSTM comprises 2 LSTMs to capture both past and future information. At each time step the hidden state at any time sequence is the concatenation of forward and backward states h_t=[$\overrightarrow{h_t^1}$, $\overleftarrow{h_t^1}$], hence the input passed to next layer is [e(w_1); h_1^1],[e(w_2); h_2^1],.....,[e(w_n); h_n^1]as the input to the next layer is the concatenation of all the previous outputs. The next layer output will be h_2 = (h_1^2, h_2^2....h_n^2). The input to the next layer will be [e(w_1); $h_1^1 h_2^1$, e(w_2); $h_1^2 h_2^2$...]. Fig. 2 shows the architure of BiLSTM.

Character-CNN: We adopted the model of [31] that leverages the one-hot encoding to build the embedding matrix for the characters to represent sequences with 256 characters. Our designed model consists of representing each character using a 27 sized vector with 26 elements for the English alphabet and one for all other symbols. This model consists of a convolution layer with kernel size 4 followed by a max-pool layer of size 3. This is fed into another convolution layer with kernel size 4 and a max-pool layer of size 3. This is followed by 2 dense layers of size 64 and 2. The strides used in convolution layers are 4 and 2.

Table 2. Details of the data set.

Data	Total	Classes	# Tokens	Test Data
D1	15476	Racism(1923); Sexism(2871) Neither(10682)	12545	CV*
D2	24783	Hate(1430); Offensive(19190) Neither(4163)	16362	CV*
D3	15001	OAG(3419); CAG(5297) NAG(6285)	15830	CV* FB: OAG(144), CAG(141), NAG(627) SM : OAG(361), CAG(413), NAG(483)

Test Data:(*CV means there was no standard train/test split and thus 5-fold CV was used). *FB* is Facebook test data and *SM* is Social Media test data.

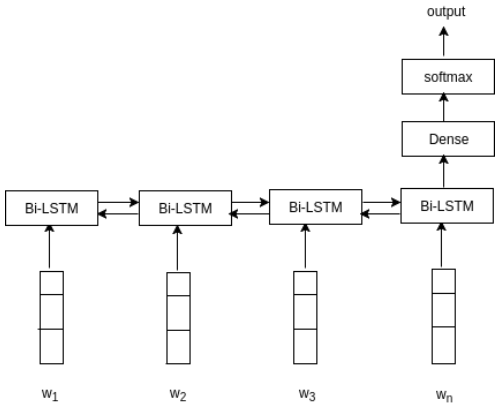


Fig. 2. Architecture of BiLSTM.

4 Data sets

For the experiments, we use three types of datasets: **D1**, **D2** and **D3**. Table 2 shows the description of all the datasets with their total instances and the number of classes.

- **D1**: This is the publicly available dataset with $\approx$ 16K Tweet IDs classified into three classes, *Racism*, *Sexism* and *Neither* by [29]. As some of the tweets were deleted as well as due to account suspension of the users we were able to retrieve around 15,476 tweets.
- **D2**: This dataset is divided into three classes *Hate*, *Offensive* and *Neither* by [9].
- **D3**: This is the aggressive data of English classified into *Overtly-Aggressive (OAG)*, *Covertly-Aggressive (CAG)* and *Non-Aggressive (NAG)* by [15]. **Table 1** shows the top occuring words in each sub-type of hate.

4.1 Experimental Setup

We use Keras [8] with Tensorflow [1] at the backend for our experiments. Experiments were performed using stratified 5-fold cross-validation to train all the classes according

Table 3. Results.

Model	D1		D2		D3	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy CV/FB/SM	F1-Score CV/FB/SM
1. CNN(W2V)	90.47	89.81	83.15	82.67	56.63/60.63/61.49	56.04/**63.14**/56.01
2. LSTM(W2V)	90.55	89.51	83.57	83.24	57.26/58.55/61.41	57.17/62.18/59.33
3. BILSTM(W2V)	90.95	89.36	83.98	83.53	58.45/55.70/58.47	58.30/59.45/58.57
4. CNN(Glove)	90.35	89.60	82.98	82.61	55.97/60.19/61.65	55.31/62.47/59.19
5. LSTM(Glove)	91.07	89.48	84.14	83.88	57.50/57.67/**64.20**	57.53/61.57/62.26
6. BILSTM(Glove)	91.08	89.99	84.25	**83.95**	58.48/55.26/59.90	58.30/59.07/59.07
7. CNN(Fasttext)	90.17	88.85	83.21	82.66	56.06/55.26/61.49	55.17/58.64/57.40
8. LSTM(Fasttext)	90.66	88.65	83.77	83.44	57.26/54.82/63.56	57.37/58.72/**62.67**
9. BiLSTM(Fasttext)	91.08	89.67	84.13	83.82	58.20/55.70/58.47	58.09/59.21/58.57
10. CharCNN	87.34	85.12	79.98	78.55	46.55/53.83/44.15	43.07/55.01/42.03
11. LSTM(Glove)+CharCNN	90.63	89.04	83.93	83.69	57.28/58.66/57.27	56.85/61.82/57.52
12. BiLSTM(Glove)+CharCNN	**91.09**	**90.39**	84.14	83.88	**58.83**/59.64/61.33	**58.72**/62.83/59.88
13. BiLSTM(Fasttext)+CharCNN	90.67	89.34	**85.86**	82.61	57.37/**60.74**/61.25	57.22/63.11/61.49
Existing State of the Art						
[29]	-	-	-	73.89	-	-
[29]	-	-	-	73.93	-	-
[5]	-	-	-	80.10	-	-
[5]	-	-	-	81.30	-	-
[5]	-	-	-	**81.60**	-	-
[9]	-	**90.00**	-	-	-	-
[10]	-	89.00	-	-	-	-
[3]	-	-	-	-	—/**62.28/61.73**	—/**64.25**/59.20
[4]	-	-	-	-	—/60.96/59.02	—/63.15/57.16
[19]	-	-	-	-	—/58.44/59.10	—/61.78/55.20
[12]	-	-	-	-	—/58.22/57.43	—/61.60/56.50
[25]	-	-	-	-	—/56.47/60.86	—/60.11/59.95
[24]	-	-	-	-	—/54.71/60.14	—/58.13/**60.09**

to their proportion. We report our results by accuracy and weighted F1-score. Categorical cross-entropy loss function and Adam optimizer were used for training because the former is very effective on the classification task than the classification error and mean square error [17]. Hidden nodes in LSTM and Bi-LSTM layers were set to 100. For regularization, dropout is applied to word embedding.

5 Results and Error Analysis

5.1 Results

Recurrent neural network based LSTM and BiLSTM performed best for all the 3 datasets. The addition of Char-CNN improved the overall accuracy and F-score. We are also discussing the existing approaches that were compared with our results in Table 3.

– Data 1

- **[29]**: Char n-grams obtained 73.89 weighted-F1 and char n-grams with gender information obtained 73.93 weighted-F1 using logistic regression classifier and 10-fold cross-validation.

Table 4. Confusion Matrix for D1(Model 6) and D2(Model 12).

Class	Dataset 1			Class	Dataset 2		
	Racism	Sexism	Neither		Hate	Offensive	Neither
Racism	1538	14	371	Hate	415	861	154
Sexism	17	1800	1054	Offensive	334	18347	509
Neither	539	441	9702	Neither	43	306	3814

Table 5. Confusion Matrix for D3(Model 12) and D3(FB(Model 1) and SM(Model 8)).

Class	Dataset 3(CV)			Class	Dataset 3(FB and SM)		
	OAG	CAG	NAG		FB/SM OAG	FB/SM CAG	FB/SM NAG
OAG	1601	1285	533	OAG	77/237	35/108	32/16
CAG	921	2842	1534	CAG	32/147	50/160	59/106
NAG	371	1531	4383	NAG	63/14	138/67	426/402

- **[5]**: Bag of words vectors(BoWV) uses the GloVe embedding with Gradient Boosted Decision Trees(GBDT), TF-IDF with GBDT and TF-IDF with SVM to obtain 80.10, 81.30 and 81.60 weighted-F1 by performing 10-fold cross-validation.

– **Data 2**

- **[5]**: Unigram, Bigrams, and Trigrams feature weighted by TF-IDF, Part-of-Speech tag unigram, bigrams, and trigrams fed into a logistic regression to obtain 90% weighted-F1.
- **[10]**: They utilized text as well as a set of metadata features to obtain weighted-F1 of 89%.

– **Data 3**

- For Data 3 the results for Facebook(FB) test data and Social media(SM) test data were being reported by various teams participated in TRAC-1.
- **[3]**: They developed LSTM and stacking of CNN-LSTM for Facebook and social media test data.
- **[4]**: The TF-IDF and latent semantic analysis (LSA) were computed for character and word n-gram features.
- **[19]**: They utilized LSTM and CNN leveraging fasttext for Facebook and social media data.
- **[12]**: SVM and BiLSTM model obtained best results for twitter and Facebook data.
- **[25]**: They combined Gated recurrent unit (GRU) with three logistic regression classifiers trained on character, word, n-grams, and hand-picked syntactic features.
- **[24]**: The designed model with a Dense architecture performs better than a Fasttext model for both social media and Facebook data.

Table 6. Metric Values.

Class	True Positive	False Positive	False Negative
Racism	79.97	26.58	20.02
Sexism	62.69	20.17	37.30
Hate	29.02	47.60	70.97
Offensive	95.60	5.98	4.40
Overtly	46.82	44.65	53.18
Covertly	53.65	49.77	46.34

Table 7. Bootstrapping Test.

Dat	Total	Sample taken	p-value
D1	15476	60%	≤ 0.01
D2	24783	60%	≤ 0.01
D3	15001	60%	≤ 0.07

5.2 Error Analysis

Error analysis was carried out to analyze the errors that were encountered in our system So we analyzed the best model confusion matrix as they were giving better performance. We did the quantitative analysis in terms of the confusion matrix and qualitative analysis for analyzing the misclassified tweets.

Quantitative analysis: Table 4 enlists the confusion matrix for Data 1 and Data 2 obtained by BiLSTM(Glove) and BiLSTM(Glove) concatenated with Character-CNN. Table 5 consists of a confusion matrix obtained by training Data 3 in cross-validation utilizing Model 12. It also contains the confusion matrix generated by testing the model with social media and facebook test data.

From Table **6** we can infer that identifying Hate, Overtly, and Covertly classes posses more challenges than other subclasses. Apart from data imbalance, using the sarcastic phrase and racial epithets in a deceitful manner makes it challenging for the classifier to identify hate sentences that had 70.97% false-negative rate and with only 29.02% true positive in D2. Due to some common obscene words between hate and offensive classes, 1.74% of offensive instances converted to hate.

Qualitative analysis: For each data set we perform qualitative analysis to analyze the errors and we find that due to hate language being contextual in nature and also when the attack is directly or indirectly on women, then the model is showing poor performance. This suggests that it is indeed difficult for models to classify into fine-grained labels. Table 8 contains some of the sentences converted to different classes due to system inefficiency.

5.3 Statistical Significance Test

We also determine whether a difference between the worst and the best classifier i.e Character-CNN and BiLSTM(GloVe) + Character-CNN is statistically significant (at p ≤ 0.05), for this we run a bootstrap sampling test on the predictions of two systems.

Table 8. Example of a sentence predicted to different class.

Data	Original	Predicted	Tweet
D1	Sexism	Neither	Please women stay single please women when you commit to your man,commit to the gym as well.
D1	Racism	Neither	@AdnanSadiq01 I think your goat is calling you. She is horny..
D1	Sexism	Racism	hate watch you have to be extra stupid to be a women and follow #Islam.
D1	Neither	Sexism	As long as she realizes she's not gonna look as pretty as she usually works.This character is a kind of mess.
D2	Hate	Offensive	@ Fit4LifeMike @chanelisabeth hoe don't make me put up screenshots of your texts to me hoe.
D2	Hate	Offensive	Offensive @vinny2vicious faggot I knew you weren't really my friend.
D2	Hate	Neither	They should have never gave a cracker a transmitter!!!!!! @realdjTV will flip when he sees this.
D3	Covertly	Overtly	D3 Covertly Overtly I told you wait.7 pak killed within hours of their cowardice act.Go and weep for them.
D3	Overtly	Covertly	yes we remember you are biggest terrorist country in the world you will do anything against humanity.

The test takes 3 confusion matrix out of 5 at a time and compares whether the better system is the same as the better system on the entire data set. The resulting (p-) value of the bootstrap test is thus the fraction of samples where the winner differs from the entire data set. Table 8 depicts the statistical significance test performed on all 3 data sets.

6 Conclusion and Future Work

In this paper we have explored the effectiveness of deep neural network for hate speech detection. The system failure on some cases highlights the subjective biases while classifying gender based message. Transfer learrning using large datasets can be very effective. Also some other linguistic features focused on gender and location will be used to improve the performance of the system. Some more other forms of hate will also be considered.

Acknowledgments. The first author would like to acknowledge the funding agency, the University Grant Commission (UGC) of the Government of India, for providing financial support in the form of UGC NET-JRF.

References

1. AAbadi, M., et al.: Tensorflow: large-scalemachine learning on heterogeneous distributed systems (2006)
2. Agarwal, S., Sureka, A.: Characterizing linguistic attributes for automatic classification of intent based racist/radicalized posts on Tumblr micro-blogging website, (2017)
3. Aroyehun, S. T., Gelbukh, A.: Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying. pp. 90–97 (2018)
4. Arroyo-Fernández, I., Forest, D., Torres Moreno, J. M., Carrasco-Ruiz, M., Legeleux, T., Joannette, K.: Cyberbullying detection task: the EBSI-LIA-UNAM system. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying. pp. 140–149 (2018)
5. Badjatiya, P., Gupta, S., Gupta, M., Varma, V.: Deep learning for hate speech detection in tweets. In: Proceedings of the 26th International Conference on World Wide Web Companion. pp. 759–760. International World Wide Web Conferences Steering Committee (2017)

6. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, vol. 5, pp. 135–146 (2017)
7. Chen, H., McKeever, S., Delany, S. J.: Abusive text detection using neural networks,
8. Chollet, F., et al.: Keras (2015)
9. Davidson, T., Warmsley, D., Macy, M., Weber, I.: Automated hate speech detection and the problem of offensive language, (2017)
10. Founta, A. M., Djouvas, C., Chatzakou, D., Leontiadis, I., Blackburn, J., Stringhini, G., Vakali, A., Sirivianos, M., Kourtellis, N.: Large scale crowdsourcing and characterization of twitter abusive behavior, (2018)
11. Gambäck, B., Sikdar, U. K.: Using convolutional neural networks to classify hate-speech. In: Proceedings of the First Workshop on Abusive Language Online. pp. 85–90 (2017)
12. Golem, V., Karan, M., Šnajder, J.: Combining shallow and deep learning for aggressive text detection. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying. pp. 188–198 (2018)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation, vol. 9, no. 8, pp. 1735–1780 (1997)
14. Kim, Y.: Convolutional neural networks for sentence classification, (2014)
15. Kumar, R., Reganti, A. N., Bhatia, A., Maheshwari, T.: Aggression-annotated corpus of Hindi-English code-mixed data, (2018)
16. Kwok, I., Wang, Y.: Locate the hate: Detecting tweets against blacks. In: American Association for Artificial Intelligence (2013)
17. McCaffrey, J. D.: Why you should use cross-entropy error instead of classification error or mean squared error for neural network classifier training (2018)
18. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
19. Modha, S., Majumder, P., Mandl, T.: Filtering aggression from the multilingual social media feed. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying. pp. 199–207 (2018)
20. Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., Chang, Y.: Abusive language detection in online user content. In: Proceedings of the 25th international conference on world wide web. pp. 145–153. International World Wide Web Conferences Steering Committee (2016)
21. Nockleby, J. T.: Hate speech. Encyclopedia of the American constitution, vol. 3, pp. 1277–79 (2000)
22. Park, J. H., Fung, P.: One-step and two-step classification for abusive language detection on twitter, (2017)
23. Pennington, J., Socher, R., Manning, C. D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
24. Raiyani, K., Gonçalves, T., Quaresma, P., Nogueira, V. B.: Fully connected neural network with advance preprocessor to identify aggression over facebook and twitter. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018). pp. 28–41 (2018)
25. Risch, J., Krestel, R.: Aggression identification using deep learning and data augmentation. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying. pp. 150–158 (2018)
26. Sharma, S., Agrawal, S., Shrivastava, M.: Degree based classification of harmful speech using twitter data, (2018)

27. Silva, L. A., Mondal, M., Correa, D., Benevenuto, F., Weber, I.: Analyzing the targets of hate in online social media. In: International Conference on Web and Social Media. pp. 687–690 (2016)
28. Warner, W., Hirschberg, J.: Detecting hate speech on the world wide web. In: Proceedings of the Second Workshop on Language in Social Media. pp. 19–26. Association for Computational Linguistics (2012)
29. Waseem, Z., Hovy, D.: Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In: Proceedings of the North American Chapter of the Association for Computational Linguistics. pp. 88–93 (2016)
30. Watanabe, H., Bouazizi, M., Ohtsuki, T.: Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. IEEE Access, vol. 6, pp. 13825–13835 (2018)
31. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in neural information processing systems. pp. 649–657 (2015)
32. Zhang, Z., Robinson, D., Tepper, J.: Detecting hate speech on twitter using a convolution-GRU based deep neural network. In: European Semantic Web Conference. pp. 745–760. Springer (2018)

ISSN 1870-4069

Event-Argument Linking in Hindi for Information Extraction in Disaster Domain

Sovan Kumar Sahoo, Saumajit Saha, Asif Ekbal,
Pushpak Bhattacharyya, Jimson Mathew

Indian Institute of Technology Patna,
Department of Computer Science and Engineering,
India

{sovan.pcs17, saumajit.mtmc17, asif, pb, jimson}@iitp.ac.in

Abstract. Event extraction is an important task in Natural Language Processing. Extracting event triggers and arguments from text is a very important sub-task of information extraction. If a sentence contains only a single event and one or more arguments, then it is obvious to assume that all the arguments are linked to that particular event. However, when a single sentence consists of multiple events and arguments, we need to link arguments with their respective events. In this paper, we develop a deep learning based approach to solve the problem of event-argument linking. We construct the task as a problem of classification, where for a given pair of event and candidate argument, the system has to decide whether they are linked to each other or not. As there is no available data in Hindi, we crawl the news data from different sources, annotate them following proper guidelines, and create a benchmark setup for event-argument linking. We believe that this is the very first attempt for event-argument linking in Hindi[1].

Keywords: Event-argument linking, deep learning, Hindi.

1 Introduction

Nowadays due to the advancement of electronic media, a massive amount of digital contents is uploaded very frequently on the internet. Extracting relevant information manually from this vast data is impossible. Information extraction concerns with developing the tools and techniques to mine the most relevant information from these data.

Event extraction is a crucial task of information extraction, used to detect the occurrence of an event along with its other details such as the time, place, agent, intensity and so on. Event mention refers to any phrase or event which describes an event. It also includes triggers and arguments. Event trigger points out the main word which highlights the occurrence of an event. Argument of an event refers to the attributes (describing the event) such as the location of occurrence of the event, time of occurrence of the event, participants involved and so on. Detection of event trigger, classification of

[1] https://github.com/Saumajit/EAL

event trigger, extraction of argument, and event-argument linking are the four important components of a typical information extraction system.

The fourth one, i.e., event-argument linking is more complex compared to the first three tasks. Generally, if any sentence consists of only one event and multiple arguments, then we can assume that all the arguments are linked to that particular event. However, if a particular sentence consists of multiple events and multiple arguments then it is difficult to decide which arguments are linked to which events. Though the task of information extraction has been explored significantly for the resource-rich language like English, this has not been the case with resource-poor language like Hindi.

One reason is the lack of availability of the annotated data for the target tasks- be it detection of event trigger, classification of event trigger, extraction of argument or event-argument linking. In our current work, we present an effective deep learning approach for event-argument linking for Hindi. We design the task of event-argument linking as a classification problem, where for a given pair of event and candidate argument, the system predicts whether they are linked to each other or not.

We use Convolutional Neural Network (CNN) [7] as feature extractor and try to classify whether there exists a relationship between an event and an argument or not. We also observe that event can lie either to the left or to the right of an argument in a sentence. Thus there exists a bidirectional relationship between an event and its arguments in a sentence. We, therefore, use Bidirectional Long Short-Term Memory (Bi-LSTM) [12] followed by CNN to address this bidirectional relationship.

In our experiment, we use Hindi news data from disaster domain. The reason behind choosing this domain is its importance and impact in our society. Extracting disaster-related information from news documents as well as from the other sources is crucial. It is useful to spread awareness among citizens and to provide relevant information to the other stakeholders such as the government departments and humanitarian agencies.

This information not only makes everyone alert but helps in overall disaster management. There is no existing dataset for information extraction in Hindi. We crawl news data from various newspapers and annotate them for our particular task. We believe that this is the very first attempt for event-argument linking in Hindi.

1.1 Problem Definition and Contributions

Given a Hindi sentence comprising of the sequence, w_1, w_2, e_1, e_2,..., e_i, w_3,...,w_k, a_1, a_2,..., a_j, w_{k+1},..., w_n, where e_i is known as an event trigger and a_j is known as a candidate argument, the task is to predict whether there exists a relationship between an event trigger e_i and an argument trigger a_j or not. Let us consider an example sentence which consists of two events and four arguments. Here, we have a total of eight event-argument pairs.

As the place argument कुलगाम (Kulgam) is linked to the event trigger बम विस्फोट (Bomb blast), we assign the classification label as '1', whereas the argument कुलगाम (Kulgam) is not linked with the event trigger आत्मघाती हमले (Suicide attack), so the classification label, in this case, is '0'. Table 1 depicts the possible event-argument pairs for the given example.

Table 1. Training instances generated from the sentence given in the above example.

Event-Argument pair	Classification label
बम विस्फोट, एक नागरिक सहित चार लोग	1
बम विस्फोट, कुलगाम	1
बम विस्फोट, छह लोग	0
बम विस्फोट, सोपियन जिले	0
आत्मघाती हमले, एक नागरिक सहित चार लोग	0
आत्मघाती हमले, कुलगाम	0
आत्मघाती हमले, छह लोग	1
आत्मघाती हमले, सोपियन जिले	1

– **Input Hindi Sentence :** कुलगाम में एक बम विस्फोट में एक नागरिक सहित चार लोग मारे गए हैं जबकि सोपियन जिले में एक आत्मघाती हमले में छह लोग मारे गए हैं ।
– **Transliteration :** Kulagaam mein ek bam visphot mein ek naagarik sahit chaar log maare gae hain jabaki sopiyan jile mein ek aatmaghaatee hamale mein chhah log maare gae hain.
– **Translation :** Four people, including a civilian, were killed in a bomb blast in Kulgam, while six people were killed in a suicide attack in the Sopiyan district.

The contribution of our current research is two-fold, *viz.* (i). We propose a deep learning based event-argument linking system in Hindi for disaster domain; and (ii). Provide a benchmark setup for event-argument linking in Hindi language.

2 Related Work

In our current work, we focus on finding the relation between an event and its corresponding argument using deep neural networks. Thus our current work falls under the lines of research of neural relation extraction. Relation extraction using deep learning technique has already been explored by the research community [16,11,14,9,13,8,10,17,15,18,2,19,6,4]. Convolutional Neural Network(CNN) is long established in relation classification. [16] suggested a CNN based relation classification approach for the first time where CNN was used to extract sentence level feature.

The features of CNN were extracted by taking all the tokens of the sentence as input, where each token was represented as the concatenation of word feature and position feature. The authors also extracted lexical level features like word embeddings of marked nouns and their context tokens and WordNet hypernyms. Both the features were then concatenated into a single vector which was then passed into *Softmax* classifier for classification. After the success of CNN in relation classification, [11] proposed a CNN based approach that performs classification by Ranking CNN (CR-CNN).

They used a novel pairwise ranking loss function that helped to diminish the impact of artificial class *Other*. [14] proposed a robust model that learns from the shortest dependency paths through a CNN. They also suggested a negative sampling strategy into their CNN model to handle relation directionality. The advantage of multiple window sizes for convolutional filters was used in [9]. Thus, their model allows the network

to capture wider ranges of *n*-grams. They also used position embedding features. A multilevel attention CNN was proposed in [13].

They used primary attention at the input level to capture entity-specific attention and secondary attention with respect to target relation for relation specific pooling attention. They claimed that their novel mechanism allows their model to detect more subtle cues of the input sentences despite their heterogeneous structure. They also introduced in this paper a novel pair-wise margin-based objective function.

Though CNN-based methods can capture high-level features, they overlooked the hierarchical and syntactical information of the input sentence. Based on this observation, the authors in [8] introduced the hierarchical layers and dependency embedding to CNN based methods to capture both the hierarchical feature and dependency structure in the window size. In a very recent work [10], a CNN based model with adversarial training method was proposed. Though CNN is very successful in capturing features for relation extraction, it captures local feature and fails to take into consideration the long-distance dependency between the nominal pairs.

To deal with this issue, the authors in [17] presented a framework based on Recurrent Neural Network (RNN). A novel neural network SDP-LSTM was proposed in [15] where they picked heterogeneous information along Shortest Dependency Path (SDP) using four different information channels. They introduced a Long Short-Term Memory (LSTM) which was built upon dependency path. To take the directionality of relation into consideration, they separated an SDP into two sub-paths where each path was from an entity to the common ancestor node.

In recent work in [18], the authors used the attention layer and tensor layer on the top of Bi-LSTM to capture word level context information and complex connection between two entities.

So far it is seen that both the CNN and RNN have been used to extract the relations. However, some researchers used the combination of both the neural network architectures to capture both the local features as well as the long distance relationship between the two entities. For example in [2], the authors used CNN on the top of LSTM units which picked up necessary information along SDP and inverse SDP at the same time through two separate channels.

In another work reported in [19], the authors used the combination of CNN and RNN along with an attention layer in between them. Apart from this, some other approaches are also reported in the literature.

In [6], authors tried to use syntax information of sentences to model the entities. They proposed to learn syntax-aware entity embeddings based on tree-GRU. They first encoded the context of entities on a dependency tree in sentence-level. Then both inter-sentence and intra-sentence attentions were used to obtain sentence set-level entity embeddings over all the sentences which contain the focused entity pair. Finally, this entity embedding combined along with a CNN based sentence embedding was used for relation extraction.

Reinforcement learning was used in [4] to deal with the noisy labeling problem in distant supervision based relation extraction methods. Their model has two modules *viz.* instance selector and relation classifier. The instance selector uses reinforcement

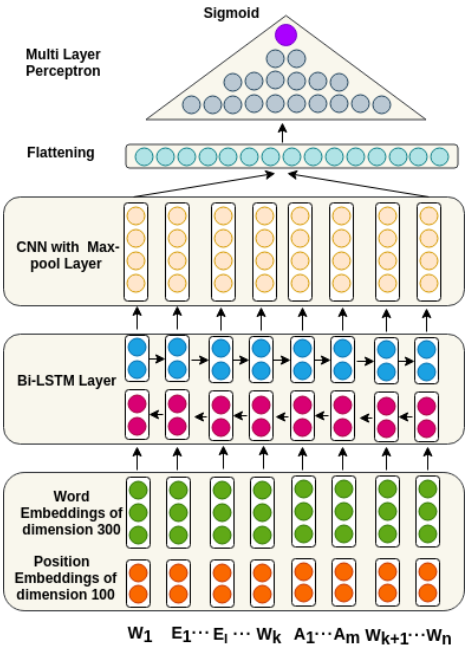


Fig. 1. The architecture of the proposed model. Here W_i denotes the words of the input sentence. E_i and A_i denote the event and argument trigger, respectively.

learning to choose the high-quality sentences and feeds the relation classifier which eventually makes the prediction and provides rewards to the instance selector.

3 Methodology

In this section, we describe the approach that we have followed for event argument linking.

3.1 Architecture

Figure 1 depicts the overall system diagram that we have used for event-argument linking. The network takes vector representation of each word of the input sentence as input and passes the vectors to a Bi-LSTM layer which captures the long term relationship between the event-argument from both the directions. The output of the Bi-LSTM layer is passed through a single-layered CNN. CNN tries to extract local convoluted features. The output of CNN is then fed into a multi-layer perceptron (MLP) model followed by a *Sigmoid* activation function for binary classification.

3.2 Input Representation

Each word w_i of the input sentence $S_i = (w_1, w_2, \ldots, w_n)$ is represented by the concatenation of two types of embeddings: (i) word embeddings (WE) which capture

Words	Four	people	including	a	civilian	were	killed	in	a	bomb	blast	in	Kulgam	while	six	people	were	killed	in	a	suicide	attack	in	Sopian	district
Relative Position w.r.t. Event	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Relative Position w.r.t. Arguments	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12

Fig. 2. Representation of the input sentence. Here each word has two relative positions with respect to Event and Argument respectively.

syntactic and semantic meaning of the word; (ii) a position embedding (PE) which identifies both the target event and arguments of our interest.

The PE also identifies the proximity of each word with respect to the target event and argument words or phrases. The input sentence S_i is the sequence of vectors $S_i = (w_1, w_2, \ldots, w_n)$, where $w_i \in R^d$ and $d = d_w + 2d_p$. d_w and d_p are the dimensions of word embedding and position embedding respectively. We choose the maximum length of each input sentence to be 100. We, therefore, use zero padding for shorter sentences and truncate the longer sentences.

3.3 Word Embedding

For word embedding (WE) of each word, we use pre-trained *fastText* [5] word vectors. These embeddings were trained on Hindi Common Crawl and Wikipedia dataset. The size of the word embedding used in our experiments is 300. The pre-trained word-embedddings are downloaded from *fastText* website[2].

3.4 Position Embedding

Position embedding (PE) was successfully applied in [16] for relation extraction. For position embedding of each word, we at first calculate the relative distance of each word with respect to event and argument trigger respectively. The relative distance can be both positive and negative. Each distance is then represented by a random vector of dimension 50.

4 Datasets and Experiments

Here in this section, we provide a description of the dataset that we have prepared for our experiments, report the results and then provide a useful analysis.

4.1 Dataset

As there was no existing dataset for event-argument linking in Hindi, we have prepared it by ourselves. The news data related to disaster events are crawled from the different news portals. All the articles are converted into XML formats and then annotated with event triggers, arguments and for event-argument linking. We have followed TAC KBP [3] annotation guidelines for our annotation task. Three annotators, with a good linguistic background, were employed for the annotation task.

Table 2. Dataset statistics. Here 'relevant instances' refer to the no of event-argument links.

Number of XML files used for training	824
Total number of relevant instances in the training dataset	7554
Number of XML files used for testing	194
Total number of relevant instances in the testing dataset	1934

Table 3. Hyperparameters used in our experiments.

Hyper parameters	**# of Epochs**	**Dropout**	**Batch size**	**# of filters**	**# of dense layer neurons**	**Dimension of WE** (d_w)	**Dimension of PE** (d_w)
Value	100	0.5	64	64	100	300	50

The tag set representing events is organized into an ontology which includes two types of events - *Natural* and *Man-made*, and ten types of arguments - *Place*, *Time*, *Casualty*, *Reason*, *Type*, *Participant*, *Intensity*, *Magnitude*, *Name* and *Speed*. The ontology has three levels where both *Natural* and *Man-made* disaster types are further divided into different sub-types. We have a total of 29 sub-types of disasters in our Hindi corpus. We measure the inter-annotator agreement ratio by asking all the three annotators to annotate 5% of total documents. The multi-rater Kappa agreement ratio of 0.85 was observed.

Table 2 shows the train-test split of total Hindi dataset. The annotated sentences are then used as input to our classification problem. Let us assume that a sentence contains two events and three arguments. We create six instances by considering each of the six event-argument pairs. For each such instance containing a particular event-argument pair, the relative distance for each word with respect to that event and argument changes. We assign the label as binary-valued (1 or 0) indicating the presence or absence of linkage between the event and argument.

4.2 Experimental Setup

For developing the system, we use the Python-based Keras [3] library with TensorFlow [1] backend. The hyperparameters are shown in Table 3.

4.3 Results and Analysis

Table 4 reports the results of all the different models in terms of *Precision*, *Recall* and *F1-Score*. Figure 3 shows the number of correctly predicted instances for both the classes for different architectures. For example, Model 3 has predicted YES (label=1) for 1068 instances where the actual label for all these instances were YES (label=1). Similarly, it has predicted NO (label=0) for 228 instances where the actual label for all these instances were NO (label=0).

Figure 3 also shows that our proposed model (Model 3) performs better than all the other models for both the classes even though *F1-score* is slightly lesser for YES class

[2] https://fasttext.cc

[3] https://www.nist.gov/tac/

Table 4. Evaluation results for event-argument linking. We report the performance of different model architectures.

	Model	Precision		Recall		F1-score	
		YES	NO	YES	NO	YES	NO
1	CNN without position embedding	0.69	0.44	0.96	0.07	0.80	0.12
2	CNN with position embedding	0.71	0.50	0.90	0.20	0.80	0.29
3	**Bi-LSTM + CNN with position embedding**	**0.74**	**0.47**	**0.81**	**0.38**	**0.77**	**0.42**
4	Stacked CNN without position embedding	0.69	0.43	0.94	0.09	0.79	0.15
5	Stacked CNN with position embedding	0.72	0.38	0.66	0.45	0.69	0.41
6	Bi-LSTM + stacked CNN with position embedding	0.73	0.42	0.74	0.41	0.73	0.42

as compared to all the other models except Model 5 and Model 6. However, *F1-score* for NO class is better than all the models and is equal to that of Model 6.

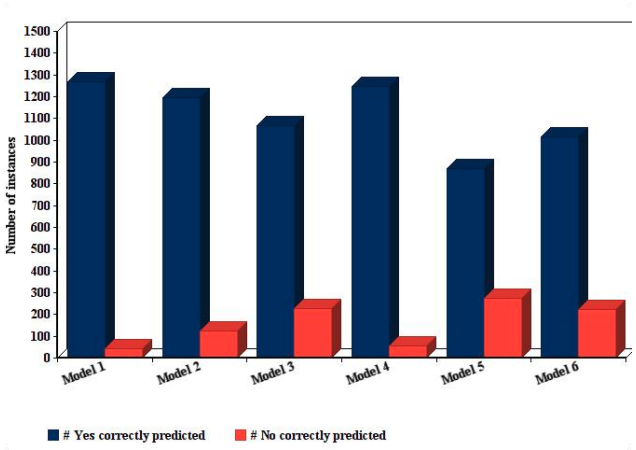


Fig. 3. Plot showing the number of correct predictions for each of the different architectures with respect to both the classes.

4.4 Error Analysis

We carry out an error analysis of the predictions of our proposed model in order to have an appropriate understanding of the system. Our analysis reveals that, out of 1934 instances in the test data, there are 625 instances for which the model has failed to correctly predict the label between the corresponding event and argument. To perform error analysis, we group the instances depending on the position of the event and argument present in the instance, i.e. whether an event lies to the left or to the right of the argument in the instance.

Based on this, we find that instances where event lies to the left and argument to the right, the system fails to detect the link in some cases when the argument

consists of numeric figures(43,727 हेक्टेयर फसल नष्ट (**Translation :** 43,727 hectares of crop destroyed). However, the system predicts the links correctly when the argument consists of a time argument in the form of(2009, 19 अगस्त 2017 (**Translation :** 19 August 2017)).

In the group of instances where event lies to the right and argument to the left, we find that the system fails to correctly predict the link involving a time argument of the type(13 दिसम्बर (**Translation :** 13 December), सुबह 8.20 (**Translation :** morning 8.20)). However, it predicts the link involving other types of argument involving numeric figures like (प्रदेश के कुल 30 जिलों (**Translation :** Total 30 districts of the state)).

This subsection shows a few of the instances where the model has gone wrong in predicting the label between the events and arguments. The word or phrase in red indicates event trigger and the word or phrase in blue indicates argument in the following instance:

1. इस बीच परवान प्रांत के प्रांतीय गवर्नर मोहम्मद असीम ने बताया कि प्रांत के दो जिलोंमें हिमस्खलनों से 16 लोगों की मौत हो गई जबकि आठ अन्य घायल हैं।
   Transliteration : is beech paravaan praant ke praanteey gavarnar mohammad aseem ne bataaya ki praant ke do jilon mein himaskhalanon se 16 logon kee maut ho gaee jabaki aath any ghaayal hain.
   Translation : Meanwhile, provincial governor of the Province Province, Mohammad Asim said that 16 people were killed and eight others were injured in avalanches in two districts of the province.
   Actual label : 1
   Predicted label : 0
   Possible reason : Place argument to the left of the event not detected.

2. 8 घंटे तक चली मुठभेड़ के बाद पाकिस्तानी सुरक्षाक्मियों ने ट्रेनिंग सेंटर पर कब्जा किया।
   Transliteration : 8 ghante tak chalee muthabhed ke baad paakistaanee surakshaakarmiyon ne trening sentar par kabja kiya.
   Translation : After 8 hours of encounter, Pakistani security forces captured the training center.
   Actual label : 1
   Predicted label : 0
   Possible reason : Participant argument not detected by the model.

3. पहला झटका सुबह के 630 बजे दूसरा झटका 645 बजे और तीसरा झटका 648 बजे लगा।
   Transliteration : pahala jhataka subah ke 630 baje doosara jhataka 645 baje aur teesara jhataka 648 baje laga.
   Translation : The first blow came at 630 in the morning, the second blow was 645, and the third shock was 648 hours.
   Actual label : 0
   Predicted label : 1
   Possible reason : Repetition of the same event twice. The model might have thought that first झटका is linked to all the three arguments.

4. भूकंप की तीव्रता रिक्टर स्केल पर 4.4 मैग्नीट्यूड मापी गई।
   Transliteration : bhookamp kee teevrata riktar skel par 4.4 maigneetyood maapee gaee.
   Translation : The magnitude of earthquake measured at 4.4 magnitude on the Richter scale.
   Actual label : 1
   Predicted label : 0
   Possible reason : Intensity argument not detected by the model.

5 Conclusion and Future Works

In this work, we have put forward a deep neural approach for event-argument linking for less-resource language like Hindi. The proposed architecture is a combination of a Bi-LSTM network followed by CNN. As there is no readily available data, we have crawled news data from the different online news sources and annotated for our experiments. The evaluation shows the promising results.

We have performed a detailed analysis of the results, and have also evaluated the effect of position embedding that shows better performance. In future, we would create more annotated data, perform event-argument linking throughout the whole document, induce coreference resolution for linking similar events, and incorporating attention mechanism for finding the best argument match for the event.

Acknowledgments. The work reported in this paper is supported by the project titled ”A Platform for Cross-lingual and Multi-lingual Event Monitoring in Indian Languages”, sponsored by IMPRINT-1, Ministry of Human Resource and Development, Government of India. Sovan Kumar Sahoo gratefully acknowledges “Visvesvaraya PhD Scheme for Electronics and IT”, under the Ministry of Electronics and Information Technology, Government of India. Asif Ekbal acknowledges Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015), software available from tensorflow.org
2. Cai, R., Zhang, X., Wang, H.: Bidirectional Recurrent Convolutional Neural Network for Relation Classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. vol. 1, pp. 756–765 (2016)
3. Chollet, F., et al.: Keras (2015)
4. Feng, J., Huang, M., Zhao, L., Yang, Y., Zhu, X.: Reinforcement Learning for Relation Classification from Noisy Data. In: Proceedings of AAAI. pp. 5779–5786 (2018)

5. Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T.: Learning Word Vectors for 157 Languages. In: Proceedings of the International Conference on Language Resources and Evaluation. pp. 3483–3487 (2018)
6. He, Z., Chen, W., Li, Z., Zhang, M., Zhang, W., Zhang, M.: SEE: Syntax-aware Entity Embedding for Neural Relation Extraction, (2018)
7. Kim, Y.: Convolutional Neural Networks for Sentence Classification, (2014)
8. Li, B., Zhao, X., Wang, S., Lin, W., Xiao, W.: Relation Classification using Revised Convolutional Neural Networks. In: Systems and Informatics, 4th International Conference on IEEE. pp. 1438–1443 (2017)
9. Nguyen, T. H., Grishman, R.: Relation Extraction: Perspective from Convolutional Neural Networks. In: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing. pp. 39–48 (2015)
10. Ren, F., Zhou, D., Liu, Z., Li, Y., Zhao, R., Liu, Y., Liang, X.: Neural Relation Classification with Text Descriptions. In: Proceedings of the 27th International Conference on Computational Linguistics. pp. 1167–1177 (2018)
11. Santos, C. N. d., Xiang, B., Zhou, B.: Classifying Relations by Ranking with Convolutional Neural Networks, (2015)
12. Schuster, M., Paliwal, K. K.: Bidirectional Recurrent Neural Networks. IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673–2681 (1997)
13. Wang, L., Cao, Z., De Melo, G., Liu, Z.: Relation Classification via Multi-level Attention CNNs. In: Proceedings of the 54th annual meeting of the Association for Computational Linguistics. vol. 1, pp. 1298–1307 (2016)
14. Xu, K., Feng, Y., Huang, S., Zhao, D.: Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling, (2015)
15. Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H., Jin, Z.: Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths. In: Proceedings of the 2015 conference on empirical methods in natural language processing. pp. 1785–1794 (2015)
16. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J.: Relation Classification via Convolutional Deep Neural Network. In: Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers. pp. 2335–2344 (2014)
17. Zhang, D., Wang, D.: Relation Classification via Recurrent Neural Network, (2015)
18. Zhang, R., Meng, F., Zhou, Y., Liu, B.: Relation Classification via Recurrent Neural Network with Attention and Tensor Layers. Big Data Mining and Analytics, vol. 1, no. 3, pp. 234–244 (2018)
19. Zhang, X., Chen, F., Huang, R.: A Combination of RNN and CNN for Attention-based Relation Classification. Procedia computer science, vol. 131, pp. 911–917 (2018)

ISSN 1870-4069

Evaluating Lightweight Text Classification Approaches for Arabic Texts

Dhaou Ghoul[1], Gael Lejeune[1], Lichao Zhu[2]

[1] Sorbonne Université,
STIH (Sens Texte Informatique Histoire),
France

[2] Université de Paris,
LLF (Laboratoire de Linguistique Formelle),
France

{dhaou.ghoul, gael.lejeune}@sorbonne-universite.fr,
lichao.zhu@gmail.com

Abstract. Epidemic Surveillance aims at detecting disease outbursts in the world in order to provide useful information to health authorities. Many automatic systems have been conceived to help these authorities to mine the available data with a special focus on press articles. The main goal of these systems is to select relevant articles by means of text classification. The secondary goal is to extract valuable information from these relevant texts . One of the main challenge is to handle many languages with different properties, various availability of language resources (lexicons, POS taggers…) and annotated data. In this paper we present a state of the art on Text Classification as well as Information Extraction for the Arabic language and we test different options for designing a lightweight system to process texts in written Arabic. We show that Arabic language has particular properties, making it difficult to handle properly without improving existing approaches. We propose improvements of an existing lightweight approach that would be promising for Arabic as well as more poorly endowed languages.

Keywords: Arabic texts, ligth text classification, process texts in Arabic.

1 Introduction

Available online press articles have become the main sources of information. The amount of published articles being increasing, it becomes difficult for users to get a comprehensive view of the data. This is why it is fundamental to have effective solutions in classifying the information to help web users find relevant documents in different areas. Among these areas figures the epidemic surveillance domain in which we need expertise of web-based epidemic intelligence systems that allow us to easily and automatically detect epidemic disease outbursts [6]. Epidemic surveillance consists of detection and interpretation of unstructured available information on the Internet. It aims to provide duly selected and indexed documents [22].

Text classification and information extraction are powerful techniques that help to structure data in order to help experts of many fields including epidemiology. The main goal of classification is to select relevant documents for a particular task whereas Information Extraction consists in extracting a structured representation from these documents in order to populate databases. Several approaches have been proposed for this task. Most of existing approaches are primarily designed for processing texts written in English, relying on sentence patterns [11], ontology-alike lexical resources [10] or hybrid approaches [17].

Extending the multilingual coverage therefore implies to reproduce a pipeline with language-dependent resources and processing tools. But this approach is not suitable for all languages [34] even with the help of machine learning [15, 30] pointed out the need to process poorly endowed languages or dialects without training data. This approach did not seem suitable for building a disease surveillance system for a language like Arabic. In this paper we will test DANIEL (Data Analysis for Information Extraction in any Language) [24], a lightweight approach which, according to the authors, allows handling numerous languages with a limited quantity of lexicon.

The system has evaluated 17 languages (English, French, Greek…) but has not, to the best of our knowledge, appropriately evaluated the Arabic language. The article is organized as follows: in Section 2, we present some works on Information Extraction and Text classification for the Arabic language; in Section 3, we describe a lightweight approach for text classification and information extraction; in Section 4 we analyze results and present some propositions to improve this approach.

2 Related Work

2.1 Event Extraction in Public Health

In Event Extraction task exits three main methods: rule-based, supervised and unsupervised method. In this section we present some related works from event extraction in public health. First, the rule-based systems based on pattern matching. For example, the French Animal Health Epidemic Intelligence System is based on a combined information extraction method relying on rule-based systems and data mining techniques [5].

Second, several supervised classifiers exist for detecting public health events within unstructured text. In these works, the authors used word embeddings such as TF-IDF, Word2Vec, etc. in order to capture relevant entity co-occurrences within a document [21]. Note that, this method is dominant in event extraction research even if it requires a large-scale labeled training corpus [36]. Finally, numerous unsupervised detection of events from text exist, like UPHED system[16].

This system identifies events as clusters of documents associated with labels, i.e., a set of diseases and locations describing clusters. This system achieves a precision of 60% and a recall of 71% using manually annotated real-world data. In [31], the authors showed that the problem of the size of ground-truth datasets when one wants to deal with more than a few languages even when the task is simplified as binary classification task like separating documents that contain epidemic events or not.

The same authors presented in [32] a dataset and a baseline evaluation for multilingual epidemic event extraction at sentence level. In this work, authors experiment with deep learning models based on a bidirectional LSTM (BiLSTM)[25] that use character and word representations and the multilingual BERT (BERT-multilingual-cased[1] and BERT-multilingual-uncased[2]) pre-trained language models for token sequential classification. BERT-multilingual-uncased achieves the F1, recall and precision scores with 80.99%, 79.77% and 82.25% respectively, on the dataset comprising relevant and irrelevant examples. Since Event Extraction needs both text classification and information extraction, we have to see these two tasks can be tackled for a language like Arabic.

2.2 Text Classification for Arabic Texts

Text classification consists of assigning unknown documents into predefined classes. The process of text classification is usually summarized in the following steps [29]:

1. Document pre-processing, i.e. tokenisation, stop-word removal, and stemming or lemmatisation,
2. Document modelling, i.e. representing a document in an appropriate form so that it can be processed by a machine learning algorithm,
3. Feature selection and projection,
4. Transforming features into classification rules,
5. Quality indicators and evaluation methods.

Regarding step 4, there are three approaches to text classification: rule-based approaches, machine learning approaches and hybrid approaches [28]. Rule-based approach organizes text into classes according to a set of handcrafted linguistic rules. If one wants to classify newspaper articles into two classes (*economy* and *health*), the easiest way would be to define two lists of tokens (usually words) that are the most discriminant for each class.

Then, when a new text needs to be classified, its class should be identified by comparing the tokens of the text to the list of the most representative tokens of each class. Here, the rules are made to select relevant tokens and to assign appropriate weights to them. For instance, in emotion detection, lexicons can be used to compute a probability to which the text belongs to a particular class [33].

These approaches can be quite simple to implement but, on the other hand, rule-based approaches have some disadvantages. These approaches require some knowledge of the domain and creating efficient language rules is expensive. Unlike rule-based approaches, machine learning approaches take advantage of past observations, rather than explicit expert knowledge, using pre-labeled examples as training data. For this purpose, the amount of training data should be sufficiently high.

[1] https://huggingface.co/bert-base-multilingual-cased
[2] https://huggingface.co/bert-base-multilingual-uncased

Table 1. Some examples of studies performing Arabic texts classification.

Reference	Corpus (# docs)	Classes	Algorithm	Accuracy	Year
Syiam et al. [35]	News (1132)	6	Rocchio	98%	2006
Duwairi [12]	Magazine/News (1000)	10	Naïve Bayes	95%	2007
Mesleh et al. [27]	News (1545)	9	SVM	98%	2008
Bawaneh et al. [7]	Unknown (242)	6	KNN	84%	2008
Ababneh et al. [1]	News (5121)	7	Cosine	95%	2014
Amina et al. [8]	News (6005)	9	SVM/Naïve Bayes	80%/ 70%	2017
Zinah et al. [2]	News (16757)	5	Master-slaves	88%	2018

The first step is to transform the text into an appropriate representation (usually vectors). One of the most frequently used approaches is bag of words, where a vector represents the frequency of a word in a predefined dictionary of words. Then, a learning algorithm (Naive Bayes, Support Vector Machine, KNN-neighbors, Deep Learning, ...) is applied, which takes a labeled learning corpus as input to create a classification model. With the appropriate amount of training data, text classification via machine learning exhibits more accurate results than rule-based approaches.

The strength of these two approaches can be combined by a process called hybridization [7]. Expert knowledge and machine learning methods are used to find a symbiosis between the simplicity of the linguistic rules and the efficiency of machine learning. Although much work has been devoted to the classification of available texts in English, Chinese and other common languages (Spanish, French …), few works have studied the classification of texts in Arabic. We will present here some of the most interesting works on Arabic texts classification.

These studies use different datasets with different algorithms but we unified the metrics to evaluate the performance of each approach presented in Table 1. A comparative description inspired by [3] and [13] is given in Table 1. We can see that most of existing works rely on machine learning approaches. As mentioned earlier, the text classification process comprises three main steps: pre-processing, classification and evaluation. As Arabic is a morphologically rich language, the pre-processing phase is crucial but we lack efficient pre-processing tools (contrary to an isolating language like English).

For Arabic texts, the pre-processing, besides tokenization and lemmatization, involves normalization of some Arabic letters. For Arabic, as well as for many other languages, there is an important gap compared to English regarding the quality of natural language processing applications. For this reason, there is a stronger interest in the scientific community towards a better treatment of both morphologically rich languages [26] and poorly endowed languages [18, 23] proposed DANIEL, a lightweight approach for Classification and Information Extraction that shows convincing results for a bunch of morphologically rich languages (Greek, Polish and Russian) and remains competitive for rather isolate languages like Chinese or English.

The rationale of their approach is to avoid classical pre-processing steps, leaving out the problems of tokenization and lemmatization, and to take advantage of text type properties. Their approach seems to be limited to news and has also been applied to Arabic and a set of languages like German, Spanish or Vietnamese but without proper evaluation.

Table 2. Precision, Recall and F-Measure of some state-of-the-art Arabic Named Entity Recognition Systems.

System	Entity	Precision	Recall	F-measure	Method	Year
TAGARAB	Number	82.8	97.0	97.3	Rule based	1998
	Time	91.0	80.7	85.5		
	Location	94.5	85.3	89.7		
	Person	86.2	76.2	80.9		
Mesfar	Number	97.0	94.0	95.5	Rule based	2007
	Time	97.0	95.0	96.0		
	Location	82.0	71.0	76.0		
	Person	92.0	79.0	85.0		
PNAES	Person	93.0	86.0	89.0	Rule based	2009
ANERsys	Location	93.03	86.67	89.74	Machine learning	2008
	Person	80.41	67.42	73.35		
	Misc	71.0	54.0	61.47		
	Organisation	84.23	53.94	65.76		
Abdulhamid/Darwish	Location	93.0	83.0	88.0	Machine learning	2010
	Person	90.0	75.0	81.0		
	Organisation	84.0	64.0	73.0		
Oudah/Shaalan	Location	x	x	90.0	Hybrid approach	2012
	Person	x	x	94.0		
	Organisation	x	x	88.0		

2.3 Information Extraction for Arabic Texts

Information Extraction goes back up to the early days of natural language processing in the 1970s. In general, Information Extraction (IE) aims to acquire knowledge from a text. Two basic information extraction tasks are named entity recognition and relationship extraction. The extraction task is carried out thanks to the filling of predefined forms. This model describes a set of entities, the relationships between them and the events involving these entities [19]. For example, a template(form) for a disease should specify fields such as: ”name of disease”, ”place of disease”, ”number of victims”.

Information extraction methods can be classified into three categories: linguistic methods, statistical methods (machine learning approaches) and hybrid methods. The linguistic methods are based on a syntactic study of text. On the other hand, statistical methods make it possible to extract information without prioritizing linguistic analysis. These methods are the most used in the processing of natural language. The hybrid methods consist of combining linguistic and statistical methods [14].

“*IE in [Arabic] poses many problems because of the morphological and graphic changesin this language: polysemy, irregular and inflected derived forms, various spellings of certain words, various writings of certain combination of characters, short(diacritics) and long vowels, most of the Arabic words contain affixes*” [20].

The Named Entity Recognition(NER) is a sub-problem of Information Extraction (IE). In the Arabic language, several systems have been created on the recognition of named entities[4]. In general, the development of an information extraction system goes through three steps: (i) identify text fragments containing information, (ii) define the structure of information representation and (iii) develop the rules to identify the information and complete the proposed form.

As mentioned previously, named entity recognition is a sub-task of information extraction. Named entities are textual elements allowing particularly relevant access to document content, that is why identifying and categorizing them is a key issue for the automatic understanding of texts. The following is a non-exhaustive list of NER tools that have been used in the Arabic NER literature.

It should be noted that the list of references provided here is not set to be exhaustive. But, to conclude, we can say that systems based on hybrid approaches have shown good performance in the recognition of Arabic named entities. But adapting these approaches for a particular task involving specific entities, like epidemic surveillance and the identification of disease names, will be costly in terms of expert time for rules creation, text annotation

3 Experiment the Lightweight DANIEL Approach

For the experiments presented here, we used the code provided by the authors of DANIEL[3]. In this section, we will present the dataset we built for this experiment and the results obtained on Arabic texts. Then, we will discuss what we have learned from these experiments and in the last part of this section we will propose some improvements to this approach and confront it to datasets in other languages.

3.1 Getting Dataset and Resources

Since the DANIEL code is designed to work with structured press articles, we managed to build a corpus of press articles in Arabic (Table 3). The method itself relies on detecting repeated character strings to perform both classification and Information Extraction. For epidemic surveillance, the authors assume that a simple list of disease names obtained from Wikipedia pages is sufficient. The rationale is that in press articles, journalists use most common words in order to ensure that the information is conveyed properly. If scientific names are used, it is only additional to these common words since the target of the press articles is mainly composed by regular speakers rather than specialists.

With this resource, the system provides a binary classification stating if a given press article is related to epidemics or not. An article is tagged as relevant if a substring S of a disease name D is found in salient positions (title, first paragraph and last paragraph) and if $\text{length}(S)/\text{length}(D) >= \theta$. θ is a threshold that can be manually tuned for each language, but the authors report that $\theta = 0.8$ provides good results in various languages. This heuristic is also used to identify the disease named entities of the document.

The definition of a relevant document may need to be better defined. It is not completely clear on how the frontier between a relevant and an irrelevant article is drawn

[3] https://github.com/rundimeco/daniel

Table 3. Statistics for the Arabic dataset, French dataset and multilingual dataset.

	ar Corpus	**fr Corpus**	**multi Corpus**
Documents	41,432	2,733	2,129
Paragraphs	$488 * 10^3$	$12 * 10^3$	$25 * 10^3$
Avg. paragraphs	$11.8(\pm 23)$	$4.5(\pm 2.3)$	$12(\pm 8)$
Characters	$97 * 10^6$	$11 * 10^6$	$5 * 10^6$
Avg.characters	$2,347(\pm 3010)$	$4,168(\pm 4,604)$	$2,402(\pm 23,99)$

since the guidelines used for human annotators[4] only indicate that in relevant articles "the main theme of the article is epidemics". Once the relevant document is detected, the system tries to locate the event at country level. There again, common names from Wikipedia are used.

If the name of a country is repeated, this country is supposed to be where the epidemics take place. Else, the location is the country where the article has been published. This rule is referred to as "implicit location". Therefore, when we built our dataset, we created a resource indicating for each particular press source, the country where it is published (see Table 3), the other two corpora have been used in [31].

3.2 Testing the Daniel Approach

Creating a reference dataset for epidemic surveillance was not our first goal since it is rather costly to build a dataset of sufficient size, so we only propose to evaluate the output of the system. This configuration does not allow us to evaluate recall, but at least we can evaluate precision. In our opinion, it is possible to verify the soundness of the approach for our purpose. Furthermore, previous research on this approach showed that the system usually produced worse results in precision than in recall.

It is a somewhat counter-intuitive statement, considering the lightness of the lexical resources. Together with the DANIEL code, we have provided a reference dataset of more than 2,000 annotated articles in French. We will also perform experiments with the reference dataset "corpus_daniel[5]" used in [23] which contains around 2,100 annotated documents in five languages. These two datasets have been annotated with the same guidelines. These datasets will be exploited to test our improvement proposals for the DANIEL approach.

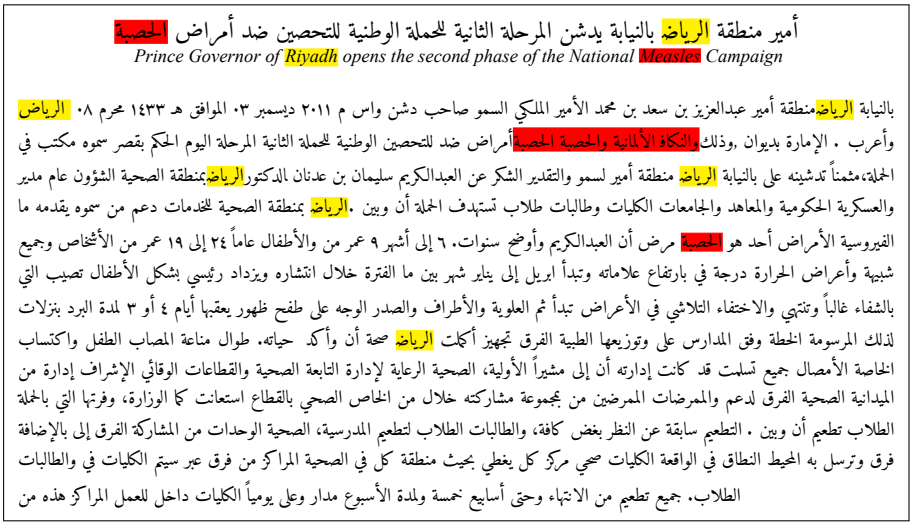
4 Results and Discussion

In this section, we present the first results obtained on the Arabic corpus. Among the relevant documents, we annotated 50 documents in order to assess precision. 54% of them were related to epidemic events.

This was not a good result and was far from what was reported in the literature for other languages. Therefore, we wanted to find out more about this result and to get an insight of the classification errors.

[4] https://daniel.greyc.fr/guidelines.pdf

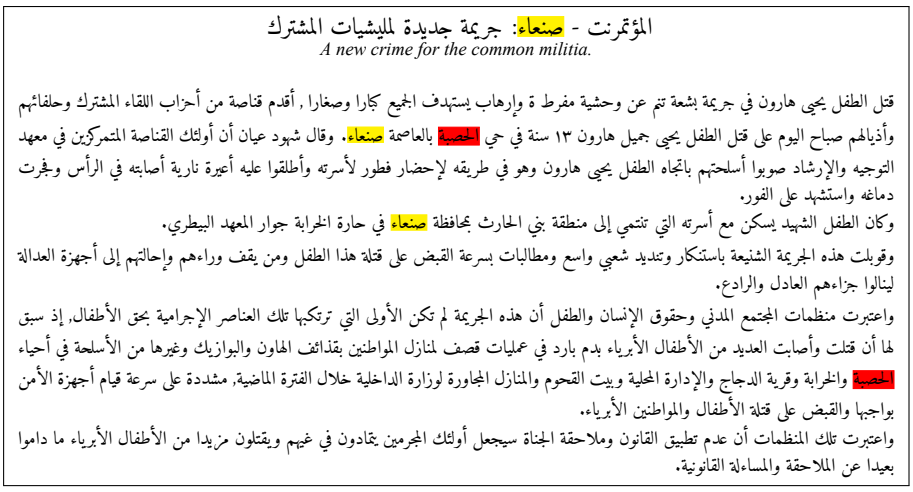
[5] https://tinyurl.com/ResearchGate-DanielCorpus



أمير منطقة الرياض بالنيابة يدشن المرحلة الثانية للحملة الوطنية للتحصين ضد أمراض الحصبة
Prince Governor of Riyadh opens the second phase of the National Measles Campaign

بالنيابة الرياض منطقة أمير عبدالعزيز بن سعد بن محمد الأمير الملكي السمو صاحب دشن واس م ٢٠١١ ديسمبر ٠٣ الموافق هـ ١٤٣٣ محرم ٠٨ الرياض وأعرب . الإمارة بديوان ,وذلك والنكاة الألمانية والحصبة الحصبة أمراض ضد للتحصين الوطنية للحملة الثانية المرحلة اليوم الحكم بقصر سموه مكتب في الحملة،مثمناً تدشينه على بالنيابة الرياض منطقة أمير لسمو والتقدير الشكر عن العبدالكريم سليمان بن عدنان الدكتور الرياض بمنطقة الصحية الشؤون عام مدير والعسكرية الحكومية والمعاهد والجامعات الكليات وطالبات طلاب تستهدف الحملة أن وبين .الرياض بمنطقة الصحية للخدمات دعم من سموه يقدمه ما الفيروسية الأمراض أحد هو الحصبة مرض أن العبدالكريم وأوضح سنوات. ٦ إلى أشهر ٩ عمر من والأطفال عاماً ٢٤ إلى ١٩ عمر من الأشخاص وجميع شبيهة وأعراض الحرارة درجة في بارتفاع علاماته وتبدأ ابريل إلى يناير شهر بين ما الفترة خلال انتشاره ويزداد رئيسي بشكل الأطفال تصيب التي بالشفاء غالباً وتنتهي والاختفاء التلاشي في الأعراض تبدأ ثم العلوية والأطراف والصدر الوجه على طفح ظهور يعقبها أيام ٤ أو ٣ لمدة البرد بنزلات لذلك المرسومة الخطة وفق المدارس على وتوزيعها الطبية الفرق تجهيز أكملت الرياض صحة أن وأكد حياته. طوال مناعة المصاب الطفل واكتساب الخاصة الأمصال جميع تسلمت قد كانت إدارته أن إلى مشيراً الأولية، الصحية الرعاية لإدارة التابعة الصحية والقطاعات الوقائي الإشراف إدارة من الميدانية الصحية الفرق لدعم والممرضات الممرضين من بمجموعة مشاركته خلال من الخاص الصحي بالقطاع استعانت كما الوزارة، وفرتها التي بالحملة الطلاب تطعيم أن وبين . التطعيم سابقة عن النظر بغض كافة، والطالبات الطلاب لتطعيم المدرسية، الصحية الوحدات من المشاركة الفرق إلى بالإضافة فرق وترسل به المحيط النطاق في الواقعة الكليات صحي مركز كل يغطي بحيث منطقة كل في الصحية المراكز من فرق عبر سيتم الكليات في والطالبات الطلاب. جميع تطعيم من الانتهاء وحتى أسابيع خمسة ولمدة الأسبوع مدار وعلى يومياً الكليات داخل للعمل المراكز هذه من

Fig. 1. An example of True Positive: Measles in Saudi Arabia.



المؤتمرنت - صنعاء: جريمة جديدة لمليشيات المشترك
A new crime for the common militia.

قتل الطفل يحيى هارون في جريمة بشعة تنم عن وحشية مفرط ة وإرهاب يستهدف الجميع كبارا وصغارا , أقدم قناصة من أحزاب اللقاء المشترك وحلفائهم وأذيالهم صباح اليوم على قتل الطفل يحيى جميل هارون ١٣ سنة في حي الحصبة بالعاصمة صنعاء. وقال شهود عيان أن أولئك القناصة المتمركزين في معهد التوجيه والإرشاد صوبوا أسلحتهم باتجاه الطفل يحيى هارون وهو في طريقه لإحضار فطور لأسرته وأطلقوا عليه أعيرة نارية أصابته في الرأس وفجرت دماغه واستشهد على الفور.
وكان الطفل الشهيد يسكن مع أسرته التي تنتمي إلى منطقة بني الحارث بمحافظة صنعاء في حارة الخرابة جوار المعهد البيطري.
وقوبلت هذه الجريمة الشنيعة باستنكار وتنديد شعبي واسع ومطالبات بسرعة القبض على قتلة هذا الطفل ومن يقف وراءهم وإحالتهم إلى أجهزة العدالة لينالوا جزاءهم العادل والرادع.
واعتبرت منظمات المجتمع المدني وحقوق الإنسان والطفل أن هذه الجريمة لم تكن الأولى التي ترتكبها تلك العناصر الإجرامية بحق الأطفال, إذ سبق لها أن قتلت وأصابت العديد من الأطفال الأبرياء بدم بارد في عمليات قصف لمنازل المواطنين بقذائف الهاون والبوازيك وغيرها من الأسلحة في أحياء الحصبة والخرابة وقرية الدجاج والإدارة المحلية وبيت القحوم والمنازل المجاورة لوزارة الداخلية خلال الفترة الماضية, مشددة على سرعة قيام أجهزة الأمن بواجبها والقبض على قتلة الأطفال والمواطنين الأبرياء.
واعتبرت تلك المنظمات أن عدم تطبيق القانون وملاحقة الجناة سيجعل أولئك المجرمين يتمادون في غيهم ويقتلون مزيدا من الأطفال الأبرياء ما داموا بعيدا عن الملاحقة والمساءلة القانونية.

Fig. 2. A False Positive example: there is a confusion because a city is named “Measles”.

We present here two examples of documents deemed relevant by DANIEL. In these two illustrations, the red colour represents the disease and the yellow colour the place of the event. The article presented in Figure 1 describes cases of measles in Riyadh (Saudi Arabia) and is, according to the guidelines mentioned earlier, a true positive. On the contrary, Figure 2 shows a False Positive: because this article speaks about a city which is called “measles”.

We can see with the True Positive example that the repetition patterns, the so-called “relevant content” algorithm, succeeds in detecting the main subject of the article. This

is not surprising as the 5W rule used in this article is also known in Arabic rhetorics[6]. In the other example, the misclassification is not due to the algorithm itself, but rather to the lexical resources it exploits.

The character string extracted is a rather small one and therefore tends to be more frequent and is prone to ambiguity. In [23] the only parameter used to avoid such False Positive cases is the θ ratio between the found substring and the lexical entry of the database. In the code published online, some corrections are made to take into account different positions in the document. But neither of these two methods is suitable to resolve the problem identified here.

Tuning the ratio would not help here, since the full string is found in the document.Modifying the relevant positions would not help either, since repetitions in first paragraph and body of the article are usually quite efficient to assess the theme of the document. Another solution might be to remove this disease name from the database, but this would surely lead to an increasing number of False Negatives. The French corpus and the multilingual corpus provided by the authors show similar False Positives cases.

In the multilingual corpus we had a False Positive case regarding "Odra" ("Measles" in Polish) because Odra is also the name of a river and the name of a small city. Another example involved the French name for "scabies" which is "gale". The substring "gale" is not uncommon in French so that some False Positives identified in the data were due to the substrings of that particular disease name. An alternative approach may be to try linguistic pre-processing and Named Entity recognition but, it would imply a paradigm shift.

So we want to find a solution that keeps the originality and the multilinguality of the original approach. We believe that the length of the disease name is the key. The longer the disease name is, the less ambiguous it is and the more confident the system should be. Setting a minimum length threshold would not fulfil the purpose. Some disease names are short, and the length would need to be tuned according to the language. The solution we propose here is to take into account not only the θ ratio but also the length of the disease name itself to compute a confidence score.

4.1 Taking into account the length of the disease names

In this configuration, we sort the documents selected by the system with respect to the length of the disease names. We take advantage of the two annotated corpora at our disposal. We observe that False Positives come mostly from short disease names. We try different configurations, first with the longest disease names ($length >= 10$) and then with names of length 9, 8 …and so on until all the disease names are introduced.

The rationale of this experiment is to see from a ROC curve how recall and precision evolve. Figure 3 shows the results obtained on the French corpus and multilingual corpus, where we can see a property of the length of the disease names. Longer disease names lead to a better precision with a recall quite low.

Shorter disease names are introduced step by step, increasing the recall but at some point with an important cost in precision. In order to have a reasonable number of False Positives, some improvements of the algorithm need to be performed.

[6] https://en.wikipedia.org/wiki/Five_Ws

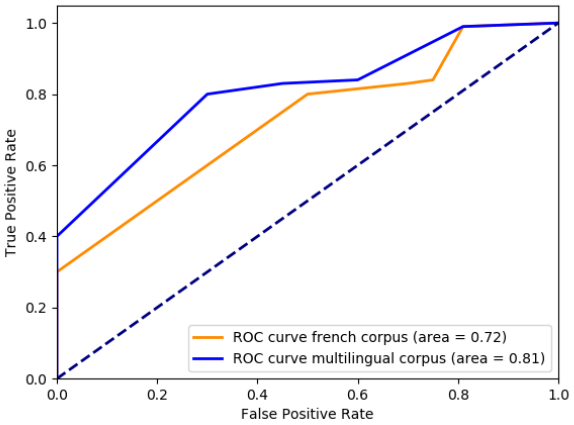


Fig. 3. ROC curve of the DANIEL system with different compositions of the lexical resource.

4.2 Discussion

We advocate that there are two ways to do this. The first one would be to use a measure to assess the potential ambiguity of the substrings found in the document. This can be done using additional lexical resources or measures like the adaptation measure [9] would surely help to improve results without setting or learning language-dependent length parameters. The other way, and more promising, would be to use long disease names to bootstrap the system by getting for each language a bunch of annotated documents with great confidence scores.

These annotated documents will then be used to learn words of the domain that are not disease names so that it would be possible to resolve ambiguities for shorter words independently of expert data. In the future, we plan to build an annotated dataset of sufficient size for Arabic to experiment with this solution, since it would help to assess how the lightweight approach for text classification and Information Extraction can be useful for Arabic texts.

References

1. Ababneh, J., Almanmomani, O., Hadi, W., El Omari, N., Alibrahim, A.: Vector space models to classify arabic text. International Journal of Computer Trends and Technology, vol. 7, pp. 219–223 (2014)
2. Abutiheen, Z. A., Aliwy, A. H., Aljanabi, K. B. S.: Arabic text classification using master-slaves technique. Journal of Physics: Conference Series, vol. 1032, no. 1, pp. 012052 (2018)
3. Al Tahrawi, M. M., Al Khatib, S. N.: Arabic text classification using polynomial networks. Journal of King Saud University - Computer and Information Sciences, vol. 27, no. 4, pp. 437–449 (2015)

4. Alruily, M., Ayesh, A., Zedan, H.: Arabic language in the context of information extraction task. In: Journal of Computational Linguistics Research. (2012)
5. Arsevska, E., Valentin, S., Rabatel, J., De Goër De Hervé, J., Falala, S., Lancelot, R., Roche, M.: Web monitoring of emerging animal infectious diseases integrated in the french animal health epidemic intelligence system. Public Library of Science, vol. 13, no. 8, pp. 1–25 (2018)
6. Barboza, P., Vaillant, L., Mawudeku, A., Nelson, N. P., Hartley, D. M., Madoff, L. C., Linge, J. P., Collier, N., Brownstein, J. S., Yangarber, R., et al.: Evaluation of epidemic intelligence systems integrated in the early alerting and reporting project for the detection of A/H5N1 influenza events. Public Library of Science, vol. 8, no. 3, pp. 1–9 (2013)
7. Bawaneh, M., Alkoffash, M., Alrabea, A.: Arabic text classification using K-NN and Naive Bayes. Journal of Computer Science, vol. 4 (2008)
8. Chouigui, A., Khiroun, O. B., Elayeb, B.: Ant corpus: An arabic news text collection for textual classification. In: 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications. pp. 135–142 (2017)
9. Church, K.: Empirical estimates of adaptation: The chance of two noriega's is closer to p/2 than p 2. In: Proceedings of the 18th International Conference on Computational Linguistics. pp. 173–179 (2000)
10. Collier, N.: Towards cross-lingual alerting for bursty epidemic events. Journal of Biomedical Semantics, vol. 2, no. 5, pp. 1–11 (2011)
11. Du, M., Von Etter, P., Kopotev, M., Novikov, M., Tarbeeva, N., Yangarber, R.: Building support tools for Russian-language information extraction. In: Habernal, I., Matoušek, V. (eds.) Proceedings of the 14th international conference on Text, Speech and Dialogue. pp. 380–387. Springer (2011)
12. Duwairi, R.: Arabic text categorization. Int. Arab J. Inf. Technol., vol. 4, pp. 125–132 (2007)
13. Elhassan, R., Ahmed, M.: Arabic text classification review. International Journal of Computer Science and Software Engineering, vol. 4, pp. 2409–4285 (2015)
14. Elsadig, M., Ahmed, A., Himmat, M.: Information extraction methods and extraction techniques in the chemical document's contents: Survey. ARPN Journal of Engineering and Applied Sciences, vol. 10, pp. 1068–1073 (2015)
15. Etzioni, O., Fader, A., Christensen, J., Soderland, S.: Open information extraction: The second generation. Proceedings of the 22nd International Joint Conference on Artificial Intelligence, pp. 3–10 (2011)
16. Fisichella, M.: Unified approach to retrospective event detection for event- based epidemic intelligence. International Journal on Digital Libraries, pp. 1–26 (2021)
17. Freifeld, C. C., Mandl, K. D., Reis, B. Y., Brownstein, J. S.: Healthmap: Global infectious disease monitoring through automated classification and visualization of internet media reports. Journal of the American Medical Informatics Association, vol. 15, no. 2, pp. 150–157 (2008)
18. Gales, M. J., Knill, K. M., Ragni, A., Rath, S. P.: Speech recognition and keyword spotting for low-resource languages: Babel project research at CUED. In: Spoken Language Technologies for Under-Resourced Languages. pp. 16–23. International Speech Communication Association (2014)

19. Grishman, R., Sundheim, B.: Message understanding conference 6: A brief history. In: International Conference on Computational Linguistics. vol. 96, pp. 466–471 (1996)
20. Hajjar, M., Zreik, K.: Classification of arabic information extraction methods. In: 2nd International Conference on Arabic Language (2009)
21. Keller, M., Blench, M., Tolentino, H., Freifeld, C. C., Kenneth, D. M., Mawudeku, A., Eysenbach, G., Brownstein, J. S.: Use of unstructured event-based reports for global infectious disease surveillance. Emerging Infectious Diseases, vol. 15, no. 5, pp. 689–695 (2009)
22. Lejeune, G., Lucas, N., Doucet, A.: Tentative d'approche multilingue en extraction d'information. In: JADT Journées internationales d'Analyse statistique des Données Textuelles. pp. 1259–1267 (2010)
23. Lejeune, G., Brixtel, R., Doucet, A., Lucas, N.: Multilingual event extraction for epidemic detection. Artificial Intelligence in Medicine, (2015)
24. Lejeune, G., Brixtel, R., Lecluze, C., Doucet, A., Lucas, N.: Daniel : Veille épidémiologique multilingue parcimonieuse. In: Proceedings of Traitement Automatique des Langues Naturelles. pp. 787–788 (2013)
25. Ma, X., Hovy, E. H.: End-to-end sequence labeling via bi-directional lstm-cnns-crf. Computing Research Repository, (2016)
26. Marton, Y., Habash, N., Rambow, O., Alkhulani, S.: Shared task system: The CADIM arabic dependency parser (2013)
27. Mesleh, A. M., Kanaan, G.: Support vector machine text classification system: Using ant colony imization based feature subset selection. In: 2008 International Conference on Computer Engineering Systems. pp. 143–148 (2008)
28. Mesleh, A. M.: Support vector machines based arabic language text classification system: Feature selection comparative study. In: Advances in Computer and Information Sciences and Engineering, Proceedings of the 2007 International Conference on Systems, Computing Sciences and Software Engineering. Part of the International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering. pp. 11–16 (2007). doi: 10.1007/978-1-4020-8741-7_3
29. Mirończuk, M. M., Protasiewicz, J.: A recent overview of the state-of-the-art elements of text classification. Expert Systems with Applications, vol. 106, pp. 36–54 (2018)
30. Munro, R.: Processing short message communications in low-resource languages. Ph.D. thesis, Stanford University (2012)
31. Mutuvi, S., Boros, E., Doucet, A., Lejeune, G., Jatowt, A., Odeo, M.: Multilingual epidemiological text classification: A comparative study. In: International Conference on Computational Linguistics. pp. 6172–6183 (2020)
32. Mutuvi, S., Boros, E., Doucet, A., Lejeune, G., Jatowt, A., Odeo, M.: Token-level multilingual epidemic dataset for event extraction. In: Berget, G., Hall, M. M., Brenn, D., Kumpulainen, S. (eds.) Linking Theory and Practice of Digital Libraries. Springer International Publishing. pp. 55–59 (2021)
33. Recupero, D. R., Dragoni, M., Buscaldi, D., Alam, M., Cambria, E. (eds.): Proceedings of 4th Workshop on Sentic Computing, Sentiment Analysis, Opinion Mining, and Emotion Detection. Co-located with the 15th Extended Semantic Web Conference, vol. 2111 (2018)

34. Steinberger, R.: A survey of methods to ease the development of highly multilingual text mining applications. Language Resources and Evaluation, pp. 1–22 (2011)
35. Syiam, M., Tolba, M., Fayed, Z., Abdel-Wahab, M., Ghoniemy, S., Habib, M.: An intelligent system for arabic text categorization. International journal of cooperative information systems, vol. 6, no. 1, pp. 1–19 (2006)
36. Zhan, L., Jiang, X.: Survey on event extraction technology in information extraction research area. In: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference. pp. 2121–2126 (2019)

ISSN 1870-4069

Improvement of a Transcription Generated by an Automatic Speech Recognition System for Arabic Using a Collocation Extraction Approach

Heithem Amich[1], Salah Zrigui[2], Mounir Zrigui[1]

[1] Université de Monastir,
Faculté des sciences de Monastir,
Tunisia

[2] L'Institut de Recherche en Informatique de Toulouse,
Toulouse,
France

heithem07@gmail.com, salahzrigui@gmail.com,
mounir.zrigui@fsm.rnu.tn

Abstract. The following study propose a novel heuristic to improve an automatic speech recognition system for Arabic language. Our heuristic relies on the collaboration of two approach: the first one ensures the extraction of collocations from a voluminous corpus then stores them in a database. It uses a combination of several classical measures to cover all aspects of a given corpus in order to exclude bigrams having a high probability of occurring together. The second one constructs a search space on the relations of semantic dependence of the output of a recognition system then, it applies phonetic filter so as to select the most probable hypothesis. To achieve this objective, different techniques are deployed, such as the word2vec or the language model RNNLM in addition to a phonetic pruning system. The obtained results showed that the proposed approach allowed improving the precision of the system.

Keywords: Automatic speech recognition, multi-level improvement, collocation, semantic similarity, phonetic pruning.

1 Introduction

Automatic speech recognition has been growing interest in recent years. It aims to facilitate communication between people and system and allows to move from an acoustic signal of speech to the transcription of the signal in a written version. Indeed, how does a transcription system work? From a recording, the system starts by calculating a transformation of the signal in acoustic parameters adapted to a

recognition engine [1]. This latter makes use of acoustic and linguistic knowledge to produce the transcription [2].

The performances of the transcription systems are good when two critical elements are well mastered, the quality of the sound recording and the availability of recordings representative of the context of use. Although an ideal transcription system remains always nonexistent, several research efforts have recently been made to come up with robust systems [3]. Automatic speech processing still has a few defect. In fact, the main limitations that hinder the development of efficient systems are generally linked to the great deal of variability in speech. On this respect, we remind of the intra-speaker variability [4], due to the elocution (singing voice, shouting, whispering, hoarse, husky, under stress), inter speaker variability (male voice, female voice, or child voice) as well as the variability caused by the signal acquisition device (type of microphone), or by the environment (noise, cross talk) [5].

Moreover, the degradation of performance is generally due to the lack of precise rules to formalize knowledge to different decoding levels (including, syntax, semantics, and pragmatics). On statistical methods with learning techniques from oral corpora where the correct transcription is known in advance. A statistical ASR is made up of several components following the acoustic and linguistic modeling of speech signal with a view to its recognition.

Many Techniques have been developed to improve each component of the system so as take account of or reduce the problems related to speech variability. Nevertheless, each technique has certain weaknesses. This leads us to develop an approach which takes account neither of the recognition modules adopted by an ASR, nor its search algorithms, or its smoothing techniques, which is the strong point of this approach. As a matter of fact, we considered the ASR as a black box device of any power of decision.

Its role is limited to providing the transcription that will trigger our correction process. Finally, our approach in the only one responsible for correcting mis-recognized hypotheses and irrelevant word [6] [7]. Also, if possible, it tries to predict the next word that speaker probably will uttered. After a brief state of the art on the technique of improving transcriptions, we describe our first approach in section 3 and the precision improvement approach in section 4, we evoke the global steps of our idea. In section 5, we integrate the concept of colocation into our system. Finally, we discuss different evaluation results. In the last section, we discuss different evaluation results in section 6.

2 State of the Art

Improving the performance of ASR caught the attention of specialists in many languages. Many works were carried out to improve the competency of the various components of the system such as the linguistic and acoustic models and to significantly improve the decoding quality and the transcription quality a priori. In this framework, Lecouteux [8] presents a combinational method allowing to exploit a priori manual transcriptions and to integrate then directly into the heart of a SARP.

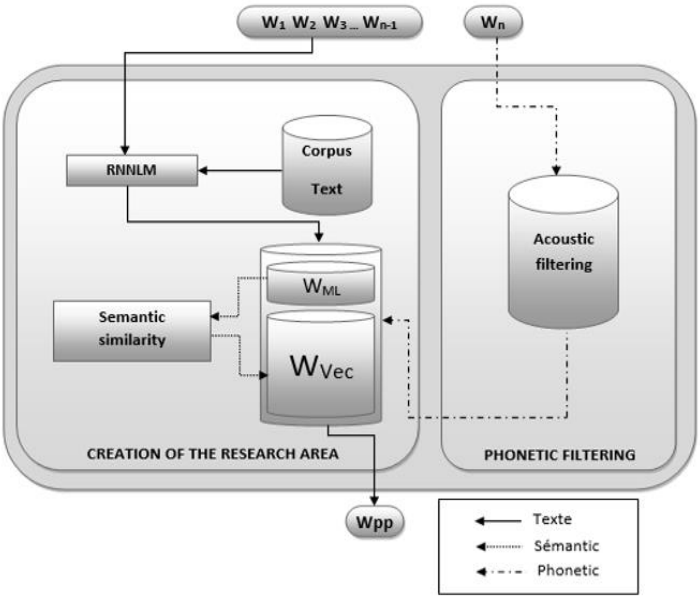


Fig. 1. The Verification and Correction System of Transcription (SyMAT).

This method allows to effectively guide the recognition system with the help of auxiliary information. He also combined SRALs based on guided decoding [9]. With reference to previous research works, Benoıt Favre [10] proposed a fusion system between an original sentence containing an error and sentence of clarification. Thus, he proposed many alignments of Levenshtein variants [11] and a reranker to select the best hypothesis. Antoine Laurent (Antoine Laurent et al, 2011) came up with a method allowing to help the user in the step of correcting ASR outputs and to correctly transcribe proper names to facilitate the automatic indexing of transcribed reunions.

Fathi Bongares [12] studied the methods of combining transcription systems of large vocabulary speech. His study focuses a on the coupling of heterogeneous transcription systems with the aim of improving the transcription quality. Combining different transcription systems is based on the idea of exploiting the strengths of each system in order to obtain a final improved transcription. In order to overcome the essential problem of natural language processing that resides in the manipulation of large volumes of texts long Med Achraf presents a collocation extraction approach based on clustering technique.

He used a combination of several classical measures which cover all aspects of a given corpus in order to drew out the consecutive pairs (w_i, w_{i+1}) of a word commonly used from a voluminous corpus. Likewise, Christopher manning exposes a number of approaches to capturing collocations such as selection of collocation by frequency or the method based on the mean and variance of the distance in more than the t-test method and mutual information.

3 Proposed Approach

In this section, we will present our system in detail. The process of automatic correction of mis- spelt words from Arabic will be done in two main phases, as shown in figure 1. The steps of the left block scheme represent the first phase. It is particularly appropriate for extending the search space for the word to correct.

The second stage is it at the right scheme. This phase is responsible for selecting the most likely word scheme.

3.1 Creation of Search Space

We expose to you the following case: the ASR has succeeded to transcribe the following word: w_0,w_1,…,w_{n-1} . By using our approach, we want to find the next word w_n badly recognized by the system ASR. The first step is to build a research space that may contain the word which we are seeking. This part is essential to develop the search space that will contain the words generated by the RNNLM language model and the semantic similarity. Let $S = w_0$, w_1,…, w_{n-1} be the context at a given instance our approach aims to estimate all of the most likely hypotheses w_n by using an RNNLM language model.

This preliminary phase consists of passing the set of observations S to a language model in order to retrieve the set of the most likely words which could complete S. The RNNLM model is based on the association of neural networks at word level. In what follows, we briefly remind of the mathematical strategies relevant to the model. Recently, deep neural networks have made a great success in the fields of image processing, acoustic modelling [13], language modelling [14] [15] etc...

Language models based on neural networks do better than standard back off n-gram models. Words are projected into low dimensional space similar words are grouped together. RNNLM could be a deep neural network LM due to its recurrent connection between input layer and hidden layer [16]. The network has an input layer x, a hidden layer S and an output layer y. We denote input to the network in time t as x (t) and output as $y(t)$. $S(t)$ refers to the state of the network (hidden layer).

In put vector (x) is formed by concatenating vector $w(t)$ which represents current word. Output is made from neurons in context layer S at time $(t-1)$ [17]. The architecture of the neural network used to calculate conditional probabilities is organized in three layers [18]. The input layer reads a word $w(t-1)$ and a continuous $S(t-1)$. The hidden layer compresses the information of these two inputs and calculates a new representation S (t) for the input of the next propagation. The value is then passed on to the output layer, which provides the conditional probabilities P (w (t) $\mid$ $w(t-1), S(t-1)$). RNNLM can be expressed as follows:

$$x(t) = w(t-1) + S(t-1), \tag{1}$$

$$S_j(t) = f\left(\sum_i U_i(t) U_{ij}\right), \tag{2}$$

$$y_k = g\left(\sum_i S_j(t) k_j\right). \tag{3}$$

Table 1. A list of Words Associated with the Word "July= جويلية " using Word2vec.

ASR	Cosine Values
June (جوان)	0.9557317
April (افريل)	0.9386088
May (ماي)	0.9097166

where $f(z)$ is a function of sigmoid activation:

$$f(z) = \frac{1}{1+e^{-z}}. \quad (4)$$

and $g(z)$ is a softmax function:

$$g(z_m) = \frac{e^{z_m}}{\sum K\, e^{z_m}}. \quad (5)$$

Semantic similarity. Identifying the similarity between words is an important TAL task regarding the domains where this technique could be useful, such as the search for information, automatic translation or even the automatic generation of text [19]. The ability to correctly identify the semantic similarity [20] between words is essential for our system [21]. This is because of its contribution to the reconstruction of research space. The search for similarity is based on the word2vec techniques [22].

Word2vec is a neural network with two layers having as an input a text corpus and as an output a set of vectors representing the characteristics of the input word in this corpus. Word is then taken to measuring the cosines similarity where an angle of 0 degree expresses a total similarity, whereas an angle of 90 degrees expresses no similarity. The following table present a list of words associated with the word «July» rising word2vec, in order of proximity [23].

Word2vec assigns a value equal to 0.6230781 to the word «France», so we deduce that France does not admit any semantic dependence with the word «July». The next step is to apply the text corpus learning and display the figure that shows the location of the words in a two-dimensional space by a projection of the main component PCA, we notice that words with the same semantic meaning are adjacent. The figure below illustrates the locations of a set of words having the same semantic context [24].

3.2 Selection of the Most Probable Word

Having collected a well-defined number of lexicons constituting the search space, we highlighted the techniques allowing filtering, classifying, and finding the most appropriate hypothesis. We adopted two filtering methods: the syntactic filtering and the phonetic, filtering [25]. *Phonetic comparison.* Having obtained a set of word W_{vec} $+W_{ML}$, we introduced another filtering mechanism operating at a phonetic level. This tool compares the frequency spectrum of the word W_n coming from an ASR and the frequency spectra of the word W_{vec} $+W_{ML}$. This method consists in aligning the signals of two words, then measuring the degree of similarity of two spectra.

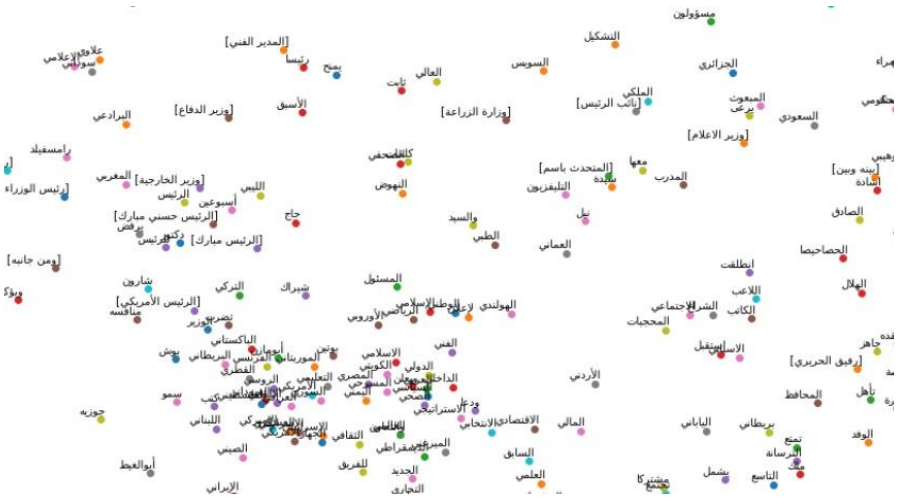


Fig. 2. The Distribution of words according to the cosine value using PCA.

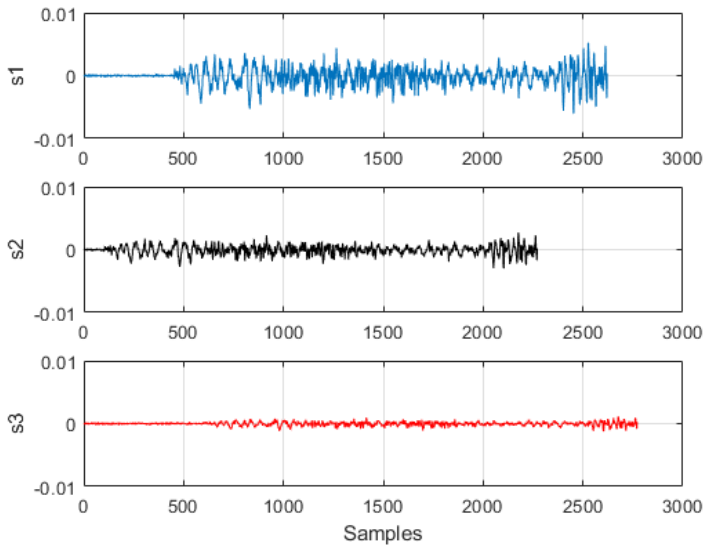


Fig. 3. Comparing the similarity of two signals.

At the end of this phase, we estimate the word W_n having the most likely label and the highest degree of acoustic similarity. This example shows how to measure the similarities of signal. Whether they are correlated or not? The black and blue signals show the signals of two most likely words generated by search space. The third signal corresponds to the word signal generated by ASR. This figure shows that there is no phonetic similarity between the two candidates with the third signal.

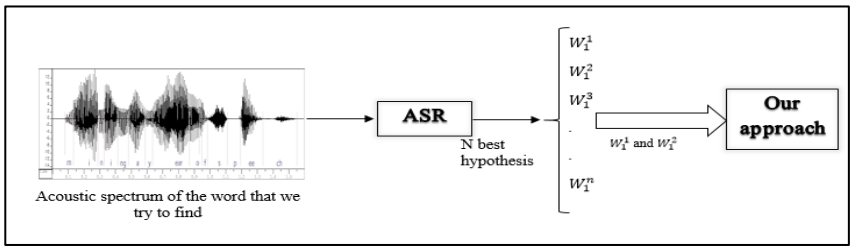


Fig. 4. First Phase of Our Approach.

Just by looking at the time series, the signal seems not to correspond to one of both models. A closer look reveals that the signals did different lengths and sample rates [26].

The case of the first word of the sentence. Concerning the previous steps of our approach, we recalled the different phases of the automatic correction of transcriptions provided by an automatic speech recognition system [27]. We elaborated architecture capable of sending back the next most likely hypothesis w_n after taking the n-1 hypotheses produced by an ASR as input: its worth mentioning that it is evident to find the words having indices between 2 and n given that there is data to manipulate.

However, at the start of our procedure, we had w_0 data to activate our approach so as to find the first word of the sentence. To overcome this limitation, we have partially changed our strategy. Indeed, we temporarily accepted the two most likely words generated by an ASR w_{11} and w_{12}. We remind that a speech recognition system uses these three pillars lexicon, the language model, and the acoustic model to provide a text representing the transcription of a sound signal (the best one). It is also possible to retain several recognition hypotheses.

The output world, then, be a list of best hypotheses N, a word graph or a confusion network. We limited ourselves to extracting the two most likely words among the retained N best hypotheses of an ASR of the first word of a sentence. This is simple due to the lack of data, which obliges us to accept w_{11} and w_{12}. However, the choice is not final. We have designed the method that reviews and verifies the first word of the sentence. The final result can accept w_{11} or rather w_{12} as well as a new lexicon retained by our approach based on a set of probabilities [28].

4 Global Steps

In this section, we will present a detailed representation of our automatic correction system of the transcript provided from a speech recognition system. This procedure is carried out in 4 steps:

– The first step consists in extracting the two best hypotheses of first word of the sentence 1 from an ASR.

– Having acquired the two hypotheses W_{11} andW_{12}, we accept W_{11}. Then, we pass W_{21} to our search approach.

  - It is essential to indicate the origin of the word. That is to say, if it is the result of the language model $W_{2\mathrm{M1}}$ or rather the result of word2vec $W_{2\mathrm{vec1}}$.

  - Of the word comes from the language model, we pass W11 and $W_{2\mathrm{ML1}}$ to our approach in order to determine $W_{3\mathrm{ML1}}$ or $W_{3\mathrm{vec1}}$. Otherwise, shift back to by using an inverse language model choose either W_{11} or W_{12} or even another word proposed by the language model. This back shift is done only when the word, retrieved by our approach, comes from the tool word2vec.

    Needless to remind that we could also define a sort of in versed language model whose words were generated in a reverse order (from right to left):

$$P_{\mathrm{reversed}}(\overrightarrow{w}) \stackrel{\mathrm{def}}{=} P(w_n)\mathrm{P}(w_{n-1}|w_n)P(w_{n-2}|w_{n-1}w_n)$$
$$P(w_{n-3}|w_{n-2}\,\mathrm{P}(w_{n-2}|w_{n-1}w_n).\,w_n).....P(w_2|w_3w_4)P(w_1|w_2w_3)$$

Following each word generated by an ASR, it is susceptible to change the old word found by our approach during a back shift. The final choice is decided when we process the last word of the sentence, which can influence or substitute the previously executed hypotheses [29].

5 Improvement Precision

In order to increase the robustness and performance of our main system shown in Figure 1 and reduce its response time. We have added a new compartment called collocation. In this section, we will present in detail the process of extraction of collocations in the system as well as the integration steps of two approaches. Collocations refer to the most widespread pair of lexemes (l_i ,l_{i+1}) commonly used in the spontaneous Arabic language. They are necessarily consecutive whose existence of a lexeme li at position X_i in a corpus T certainly requires the presence of the lexeme l_{i+1} at the position X_{i+1}.

A collocation is expression of two words that corresponding to some conventional method of saying things. There is considerable overlap between the concept of collocation and notion like term, technical term and terminological. Collocation are crucial for several domain: natural language generation, computational lexicography and corpus linguistic research. It comprises:

– Proper names: الولايات المتحدة (United States)

– Verbal expression: (I saw the light) أبصر النور

– Terminologies: (Hello) السلام عليكم

5.1 Conventional Approaches Extracting Collocations

The t test. If two words occur together many times, then we expect the two words to co-occur a lot just by chance. The t- test has been widely used for collocation discovery. It looks at the difference between the observant and expected means. If t is large enough the w1 and w2 are associated, we compute the t static:

$$t = \frac{\bar{X} - \mu}{\sqrt{\frac{S^2}{N}}}. \tag{6}$$

where $\bar{X}$ is the sample mean, N is the sample size, μ is the mean distribution and S^2 is the sample variance [20]. *Likelihood ratio.* is further method for hypothesis testing. In applying this test to collocation discovery, we have the ability to distinguish the occurrence of both common and rare phenomenon [20]. This method gives two hypotheses and test which one is most probably, the two hypotheses $H_1$and H_2 are:

- H_1: independence between w_1 and w_2: $p(w_2|w_1) = p(w_2|\neg w_1) = p$.
- H_2: dependence between w_1 and w_2: $p(w_2|w_1) = p_1 \neq p_2 = p(w_2|\neg w_1)$.

The likelihood ratio is:

$$\lambda = \frac{L(H_1)}{L(H_2)}. \tag{7}$$

where L is the likelihood function, assuming a binominal distribution L is given by:

$$L(p; n, r) = r^p (1 - r)^{n-p.} \tag{8}$$

where n is the number of trials, r the number of successes, and p is the probability of success. *Mutual information*. is a measure of how much one tell us about the other. It allows to compare the probability of observing w_1 and w_2 independently $p(w_1)p(w_2)$ mutual information is calculated by:

$$I(w_1|w_2) = \log_2 \frac{p(w_1|w_2)}{p(w_1)\, p(w_2)}. \tag{9}$$

If mutual information is large then w_1 and w_2 are related else, it is too low then w_1 and w_2 are independent [20].

5.2 Steps of Extraction of Collocations

To extract all the most common collocations of the arable language, we have combined the three methods recently mentioned, called the t-test, the Likelihood ratio and the mutual information. Thus for each candidate of the collocation w_1 w_2, these thee measures will be used to calculate the dependency between w_1 and w_2. Then, we calculate the average value of the three measures for each bigram.

We consider a collocation all bigrams corpus having a mean higher than a very high empirical threshold. The preliminary step consists of segmenting the corpus by identifying the basic units forming the corpus. This means identifying the separators

used to isolate the morphemes. We also define a stop list to omit the words that cannot form a collocation as:

– The particles of coordination: (ثم، أم، أو، أما، إما).

– The interrogative particles: (أيّ, كيف, أين, متى).

– The particles of Appeal: (يا، أيا ،أيها هيا).

Once the bigrams have been identified, the next step is the calculation, for each bigram, we calculate the average of three measures mentioned previously. If the value found is greater than a threshold, then the bigrams is considered collocation and we add it to the list of collocations. The figure below illustrates some of the accumulated collocations in our database collocation [30]. Notations used are summarized in the following:

– T: Corpus size.

– L_i : lexeme i, 1≤ i ≤ T .

– Bi : Bigram i.

– SL : Stop List.

– E_i : a real which designates the calculated average of each bigrams.

```
1. //Bigrams extraction and Measures computation
2. For all lexemes l_i, 1 ≤ i ≤ T − 1  Do
3. B_j = { l_i, l_{i+1} / l_i ∉ SL ∧ l_{i+1} ∉ SL } End.
4. //calculate the average of each bigram
5. Ei = average (Mutual inf (Bi), t test (Bi), Likehood.ratio(Bi)
6. //add the bigram to all the collocations If (E_i > thrushold) {
7. Col = Col ∪ w_i, w_{i+1} }. End
```

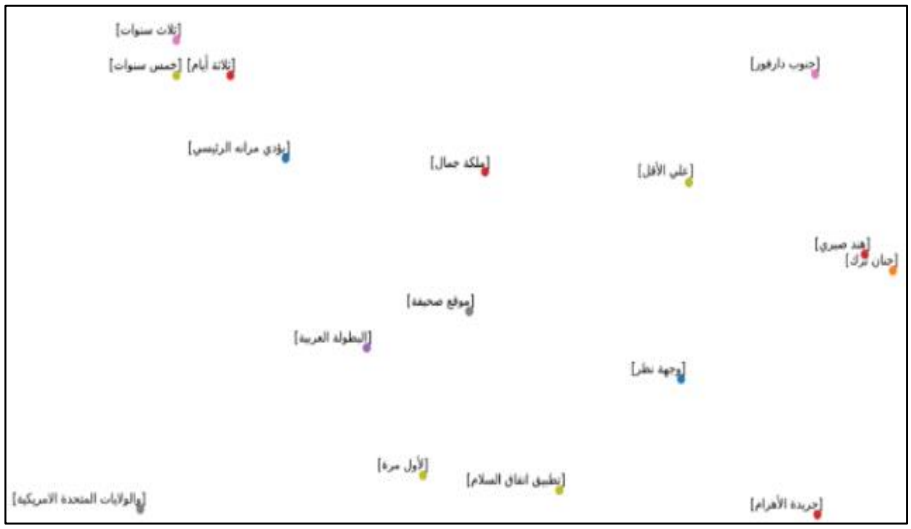


Fig. 6. Colocation Group in a Two-Dimensional Space.

Table 2. Results of the system.

ASR	Precision	F-measure
Sphinx	51,38	56,41
Sphinx + SyMAT	56,52	62,05
HTK	46,24	50,77
HTK + SyMAT	52,72	57,88

5.3 Integration of Collocations into the Main System

At this level, we have completed the construction of a collocation base. However, the obvious question is about the contribution of integrating the concept of collocations into our system? Let S=w_0,w_1,…,w_{n-1} be the context at a given instance. S represents the words pronounced by the speaker. At this point, our system has completed the verification of the whole sequence in order word after word with success. Let w_n be the word that will be treated. If the word w_{n-1} does not belong to the stop list, our heuristic checks if the previous word w_{n-1} is part of one of the collocations previously collected.

We recall that a colocation is composed of two lexemes (l_i,l_{i+1}). If w_{n-1} exists in the collocation base (w_{n-1}=l_i), then it is sufficient to apply the acoustic comparison of w_n with the second lexeme l_{i+1}. That means that the steps for creating the search space provided by the RNNLM language model and word2vec be canceled, the general heuristic has two paths, if the last word processed by the system is part of the collocations, then we just perform the acoustic test.

If this test is positive, then this is the word to look for. If not, we execute as usual our initial approach (SyMAT).

Table 3. Samples of collocation candidates.

w_1w_2	M.I	T.T.	L.R.	E_i
مليون دولار (Million dollars)	1	1	1	1
أشراط الساعة (Signs of the Hour)	0.999	0.975	0.877	0.95
الطبعة الاولى (First Edition)	0.649	0.255	0.346	0.416
اتفاق السلام (Peace Agreement)	0.781	0.79	0.857	0.809

The integration of the colocation approach into the SyMAT system is very beneficial to the level of confidentiality and accuracy of the final result. Indeed, if the word belongs to the list of collocations stored, and the acoustic test established is positive. Doubtless, we are confident that this is the exact word uttered by the speaker.

6 Experimentation

To construct the language model, we have used an Arabic text corpus of 100M words collected from corpus available on the used. This same corpus served to the construction of the model based on label. As for the testing of our system, we recorded a caustic corpus of 40 hours. We set up our SyMAT system at the exit of two known SPAP namely Sphinx [24].

The obtained results show that the proposed approach effectively contributed to improving ASR. We may also note that our method is more efficient for the HTK system than for the Sphinx system. This is justified by:

- The high clean error rate of the HTK system as compared to the sphinx system [31].
- The acoustic models trained by Sphinx were much better than that of HTK [32].

The obtained results show that if the sum of the three values exceeds a threshold equal to 0.8, then the bigram is considered collocation.

7 Conclusion

On this paper, we propose a heuristic with the aim of improving the transcription generated by an ASR for Arabic. This method exploits semantic, phonetic levels and collocation's concept in order to evaluate the output of the ASR system and to propose the most likely word in case there is an error. To enforce this approach, we resorted to

the techniques of word similarity, t test, mutual information, likelihood ration and to the RNNLM language model to establish a search space based on the history of a transcription $W_1 \dots W_{n-1}$

After that, we carried out a phonetic pruning to choose the most probable word. We also resorted to the techniques of t test, mutual information, and likelihood ration to extract collocations in order to increase the exactitude of final result. As a future work, we hope to promote our system from a model allowing taking account of the historic of applied corrections and assuring adaptation of the correction process to a particular user.

Reference

1. Mohit-Dua, R. K., Aggarwal, V.K., Shelza, D.: Punjabi automatic speech recognition using htk (2012)
2. Rajesh-Kumar, A., Mayank, D.: Acoustic modeling problem for automatic speech recognition system: advances and refinements (part II). I. J. Speech Technology, vol. 14, no. 4, pp. 309 (2011)
3. Andrs-Zolnay, R. S., Hermann, N.: Robust speech recognition using a voiced-unvoiced feature.
4. Matthew, A., Richard M. S.: On the effects of speech rate in large vocabulary speech recognition systems. In: International Conference on Acoustics, Speech, and Signal Processing, ICASSP '95, Detroit, Michigan, USA, pp. 612–615 (1995)
5. Sherry Y. Z., Suman, V. R., Nelson, M.: Multi-stream to many-stream: using spectro-temporal features for ASR. In: INTERSPEECH 2009, 10th Annual Conference of the International Speech Communication Association, Brighton, United Kingdom, pp. 2951–2954 (2009)
6. Rohit, P., Rohit, K., Sankaranarayanan, A., Wei, C., Sanjika, H., Matthew, E. R., Fred-Choi, A. C., Enoch, K., Arvind, N., Prem, N.: Active error detection and resolution for speech-to-speech translation. In: International Workshop on Spoken Language Translation, IWSLT 2012, Hong Kong, pp. 150–157 (2012)
7. Marin, A., Kwiatkowski, T., Ostendorf, M., Zettlemoyer, L.: Using syntactic and confusion network structure for out-of-vocabulary word detection. In: IEEE Spoken Language Technology Workshop (SLT), pp. 159–164, (2012) doi: 10.1109/SLT.2012.6424215.
8. Benjamin, L., Georges, L., Stanislas, O.: Integrating imperfect transcripts into speech recognition systems for building high quality corpora. Computer Speech and Language, vol. 26, no. 2, pp. 67–89 (2012)
9. Benjamin, L., Pascal, N., Georges, L.: Semantic cache model driven speech recognition. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, Sheraton Dallas Hotel, Dallas, Texas, USA, pp. 4386–4389 (2010)
10. Benoit, F., Mickael, R., Frédéric, B.: Reranked aligners for interactive transcript correction. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, Florence, pp. 146–150 (2014)
11. Tor, H., Torleiv, K., Vladimir, I.: Error correction capability of binary linear codes. IEEE Trans. Information Theory, vol. 51, no. 4, pp. 1408–1423 (2005)
12. Mars, M., Antoniadis, G., Zrigui, M.: Statistical part of speech tagger for Arabic language. In: Proceedings of the 2010 international conference on artificial intelligence.
13. Bougares, F., Esteve, Y., Deléglise, P., Linares, G.: Bag of n-gram driven decoding for LVCSR system harnessing. In: 2011 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2011, Waikoloa, HI, USA, pp. 278–282 (2011)

14. George, E. D., Dong-Yu, L. D., Acero, A.: Context dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Trans. Audio, Speech & Language Processing, vol. 20, no. 1, pp. 30–42 (2012)
15. Ebru, A., Tara, N., Brian, K., Bhuvana, R.: Deep neural network language models. In: Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT, WLM '12, pp. 20–28 (2012)
16. Tomas, M., Martin, K., Lukas, B., Jan, C., Sanjeev, K.: Recurrent neural network based language model. In: INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, pp. 1045–1048 (2010)
17. Pennington, J., Socher, R., Manning, C. D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, pp. 1532–1543 (2014)
18. Manning, C., Schutze, H..: Foundations of statistical natural language processing. Cambridge, Mass. MIT Press (1999)
19. Mahmoud, A., Zrigui, M.: Deep neural network models for paraphrased text classification in the Arabic language. In: International Conference on Applications of Natural Language to Information
20. Batita, M.A., Zrigui, M.: The enrichment of Arabic Wordnet antonym relations. In: International Conference on Computational Linguistics and Intelligent Text Processing
21. Frédéric, B., Benoit, F.: ASR error segment localization for spoken recovery strategy. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, pp. 6837–6841 (2013)
22. Mohamed-Achraf, B.M., Souheyl, M., Mohamed-Amine, N., Mounir, Z.: Exploring the potential of schemes in building NLP tools for Arabic language. International Arab Journal of Information Technology, vol. 12, no. 6, pp. 566–573 (2015)
23. Batita, M.A., Zrigui, M.: The enrichment of Arabic Wordnet antonym relations. In: International Conference on Computational Linguistics and Intelligent Text Processing.
24. Mallat, S., Zouaghi, A., Hkiri, E., Zrigui, M.: Method of lexical enrichment in information retrieval system in Arabic. International Journal of Information Retrieval Research (IJIRR), vol. 3, pp. 4, pp. 35–51
25. Zouaghi, A., Zrigui, M., Antoniadis, G.: Compréhension automatique de la parole arabe spontanée. Traitement Automatique des Langues, vol. 49, no. 1, pp. 141–166
26. Mahmoud, A., Zrigui, M.: Semantic similarity analysis for paraphrase identification in Arabic texts. In: Proceedings of the 31st Pacific Asia conference on language, information and computation
27. Ayadi, R., Maraoui, M., Zrigui, M.: Latent topic model for indexing Arabic documents. International Journal of Information Retrieval Research (IJIRR), vol. 4, no. 2, pp. 57–72
28. Boudhief, A., Maraoui, M., Zrigui, M.: Call system based on pedagogically indexed text. In: Palestinian International Conference on Information and Communication Technology
29. Merhbene, L., Zouaghi, A., Zrigui, M.: A semi-supervised method for Arabic word sense disambiguation using a weighted directed graph. In: Proceedings of the Sixth International Joint Conference on Natural Language Processing
30. Bsir, B., Zrigui, M.: Enhancing deep learning gender identification with gated recurrent units architecture in social text, Computación y Sistemas, vol. 22, no. 3, pp. 757–766 (2018)
31. Satori H., Hiyassat H., Harti M., Chenfour N.: Investigation Arabic Speech Recognition Using CMU Sphinx System. The International Arab Journal of Information Technology, vol. 6, no. 2 (2009)

ISSN 1870-4069

Comparative Study of Text Summarization Approaches

Amina Merniz, Anja Habacha Chaibi, Henda Hajjami Ben Ghezala

University of Manouba,
RIADI Laboratory, National School of Computer Science (ENSI),
Tunisia

{anja.habacha,henda.benghezala}@ensi.rnu.tn
amina.merniz@ensi-uma.tn

Abstract. Since 1958, automatic text summarization has been an interesting field that emerged to be more effective considering the exponential growth of information on the web. An intensive research on the matter of this subject has been conducted last year. Text summarization is the process of generating concise summary of the original text by reserving the main ideas and the relevant information. The paper presents the different types of automatic text summarization namely: (single document or multi-document), (generic or request oriented), (Abstractive or Extractive), (monolingual , multilingual or cross-lingual), in addition to the classification of techniques used to produce a summary, including limits and advantages of the related works in each category has been covered. A comparative study of recent text summarization systems has been introduced, which demonstrates that most of the studies focused on extractive text summarization, however summaries in the Arabic are still limited. In addition the thematic aspect was not taken into consideration. The objective of the paper is to help researchers to concentrate on those limitations to decide their future directions.

Keywords: Automatic text summarization, extractive, abstractive, monolingual, multilingual.

1 Introduction

In the last years, we live in a huge increase in the amount of data, which allows the data science to explore them also to extract knowledge. This continuous growth of information on the web requires developers to search for a way to retrieve information even to provide accurate and complete idea of document content. Hence they are oriented towards the construction of automatic document text summarization started with the generation of single document text summarization Luhn (1958) [1], then passed to multi-document text summarization [5, 32]. Several techniques have been generated in this area until now. An automatic text summarization should be relevant, concise, and shorter than the original text. For multi-document text summarization there are some issues to be avoided such as redundancy, meaning, sentence order, etc.

Thereby making automatic text summarization still challenging task. The remainder of paper is presented as follow: Section 2 defines different types of text summarization. Section 3 describes classification of related works according to the techniques used in text summarization with limits and advantages of each related work. Section 4 discusses the comparative study of these approaches. Finally, the conclusion is presented in Section 5.

2 Types of Text Summarization

We define summary as one of the crucial tasks in Natural language processing. Which consists in abbreviating one or several texts in a shorter version, by reserving only the main ideas and the most relevant information. In order to facilitate the task of reading to the user, as well as help him to obtain a clear, meaningful, and accurate view about the content of the source text. We classify a summary depending on the following categories: based on documents numbers (single or multi-document), based on summary usage (generic or query request), based on characteristics of summary as text (abstractive or extractive), based on language (monolingual, multilingual or cross-lingual).

Based on the number of documents: single document (**S**) it takes only a single document as input to summarize and also generates a single document. Multi-document (**M**) it takes as input a collection of documents, produces the summary then output a single document. Based on summary usage: generic summary (**G**) The summary produced based purely on the text source. Query-oriented (**O**) the summary is guided by a user's request.

In that case, the system must first locate on the set of documents the passages involved with the user request, then produces the summary [2]. Based on the characteristics of summary as text: abstractive summary (**A**) the term abstractive is used to describe a summary, that requires a detailed study of the text. It can synthesize a short version of the original sentence even to add new vocabulary or a new sentence not included in the final source text.

The objective thus of the abstractive summary is to reduce the redundancy and improve the compression ratio [3]. Extractive summary (**E**) Most studies have based on the construction of extractive summary because it avoids the problem of generating the text which is always regarded as(considered) a very complex task. It is based on statistical analysis to assign scores to original text sentences to extract the most frequents ones to produce the summary [3].

Based on language: Monolingual (**ML**) the language of the source document and the target document are identical. Multilingual (**MUL**) The source document is in different languages (English, Arabic, French or others), the summary is also generated in these languages. Cross-lingual (**Cross**): The source document is in the language while the generated summary is in another language different from that of the source text.

3 Techniques of Text Summarization

3.1 Statistical Based Approaches

This approach is simple, it consists of extracting keywords from text documents. Based on statistical features such as TF (Term Frequency), TF-IDF (Term Frequency-Inverse Document Frequency), POK (Position of Keyword), and others. Its principle is to compute a score characterizing the importance of each textual unit (sentence, paragraph, ...), then retain units that have a score above a certain threshold, until reaching the rate or percentage usually defined by the user [25]. Fukumoto (2004) [5], proposed a summary system which automatically classifies documents into three types: "one topic type", "multi-topic type", or "others".

Both single, also multi-document summary, in the first case, it is based on TF-IDF and sentence position in the document to assign weight to sentences and extract those with higher scores and eliminate non-important sentences. In the case of multi-document summarization, it exploits and reapplies the technique used for the single-document summarization for each document then generates the final summary. This system uses a simple strategy to create the summary.

The limit of this work is that the implementation involves some system bugs in the classification mechanism. Ouyang et al. (2009) [6], create a new hierarchical tree representation of words based on the most frequent terms at the top of the hierarchy. They estimate the words sense on the tree and extract the sentences able to integrate various objectives of multi-document to generate a relevant summary. The benefit of the generated system is that the idea of incorporating goals of multi-document summary into one framework is meaningful.

The disadvantages of the system are that it fails to create better summaries on some document sets, the constructed hierarchical tree can't always represent the concepts for some documents sets, another problem is that the two constraints used in the algorithm of tree construction are still not correct in the real data. Gupta et al. (2012) [7], present a statistical text summarization approach using the kernel of the original text. The kernel-based system, called Kernel- Sum (KERNEL SUMMarizer), uses the kernel as a guide or guideline for identifying and selecting segments of text to be included in the summary.

The determination of kernel sentences relies on simple statistical methods namely: kernel Key based on the identification of keywords, and kernelTFISF based on the inverse distribution of sentences in the source text. The size of the summary can be specified by the user when calling the tool. The proposed approach includes the kernel. The extracts convey well the main idea of the source texts. The inconvenient of the proposed summary is that many authors have stressed the need to convey the main idea and justification of the results in the automatic summary.

El-Haj et al. (2013) [24], presented a general, extractive, multilingual summary for both types of summary single as well as multiple. They used the same pipeline processing for both summaries. A log-likelihood score is computed according to dataset words. Log-likelihood helps to identify words that are frequently unexpected. Summaries are constructed by selecting the highest sentences sum of the log-likelihood scores of their words.

In this work, single-document summaries, in English and Arabic, have worked particularly well in the automatic evaluation. They are ranked first and second respectively. The limitations consist the low scores of automated assessment of Arabic and English multi-document summaries due to two main reasons. First, they treated all the multiple documents as one big document. Second, they did not work to eliminate redundancy. Finally, the log likelihood score could be improved by the inclusion of a scatter score or weight to examine the regularity of the spread of each word in all documents. Bhatia et al. (2015) [25], proposed system for single document summarization.

The source document is converted to a Universal Networking Language (UNL) document by the En-Conversion process. The textual summary algorithm is applied to the UNL document. This summary document is provided to EUGENE for conversion to a different natural language. The proposed algorithm consists of deleting the relations that are not important to minimize the complexity, Then calculate the score of each UNL sentence. Each UNL sentence consists of words and universal connection ships. The sentence score is the total of universal word weights, the weight of these words is computed using TF-IDF.

Then sentences are chosen according to their calculated score and the document size, the phrases having the high score are included in the summary. After deleting the redundancy in the selected sentences, they combine those who have the same universal word as the head to form a complex and meaningful sentence. Finally, the summary obtained is processed again before it is passed to DeConverter for further processing. The limitation of the system is that the algorithm has not been applied to the multilingual document, which makes its performance weaker and unjustified.

3.2 Machine Learning Based Approaches

Approaches of machine learning are organized into classes namely : supervised, unsupervised, or semi-supervised. The first category has a set of documents and their corresponding references generated by humans. In summarization sentence is labeled as correct when it belongs to the reference summary, or as incorrect in the opposite case, using a collection of learning documents the most popular techniques used in this category are: Nave Bayes classification, mathematical regression, neural networks.

In the second class, systems do not involve training data. They produce the final summary based only on the target documents. They determine the hidden structure in unlabeled data. Thus, they are suitable for all newly observed data without any advanced modification. Such systems are guided by heuristic rules to select very relevant sentences to be included to the summary, Clustering, Markov's model, etc. are examples of unsupervised techniques, for the last category, it requires labeling or/and unlabeled data to generate an appropriate classifier.

Agarwal et al. (2011) [9], the summary system is called Scisumm, it applies Texttiling algorithm to segment the input documents. Then generate the labeled clusters from the segments, using an algorithm based on clustering and various terms. The groups are classified according to relevance to the query generated by the user. This is achieved by a ranking module that uses the cosine similarity between the question and the centroid of each cluster.

Finally, the blade of summary generation display the groups obtained. The approach is based on clustering which is extremely fast in execution time, so it is relatively efficient regarding space required gives ends, a set of many names is generated for each cluster which a comprehensible cluster description. The inconvenient of the system is that it creates summaries of scientific articles. The order of groups based on relevance must be improved to make the set of observations of the cited articles more diverse. Li et al. (2007) [10], extract sentences from the original text and reorganize them into a summary taking into consideration the query.

The process of extracting sentences depends on various characteristics; the SVR (Vector Support Regression) approach is used to combine these characteristics. We note that the Vector Support Regression (SVR) template is used to connect features and mark sentences automatically, the appropriate lexical and syntactic characteristics are adopted and SVR appropriately assigns the weight parameters. The limitations consist that many sentence simplification and reorganization methods are not introduced, the performance in responsiveness evaluations is not good.

Yong et al. (2011) [11], based on neural networks to generate summary in three steps: A text preprocessing subsystem, a keyword extraction subsystem, a summary production subsystem. The benefit of the proposed system is that the competitive network architecture of the system is carefully designed as it directly affects the suitable system's output. In contrast, the system does not cover all kinds of documents such as: legal documents, and financial, also the considered system generates the summary without compromising readability.

Nallapati et al. (2016) [12], present SummaRuNNer Recurrent Neural Network Encoder-Decoder based on sequence model for text summarization. In which they propose models for construction summary problems such as: capturing keywords, modeling rare or invisible keywords, and the capture of document structure hierarchical. This article offers several solutions for text summarization problems many of these solutions have contributed to improved performance.

While the model is misinterpreting the semantics of the text, capturing the meaning of complex sentences remains a weakness for this model, the same phrase is often repeated in summary. Fejer and Omar (2014) [26], propose single and multi-document Arabic text summarization, based on clustering algorithms namely: the agglomerative hierarchical classification algorithm with (single link and complete link) and k-means. Then extract key phrases from each cluster, reorganize and classify them.

The similarity algorithms, namely the cosine similarity and the Jaccard coefficient, are used to select a sentence from each set of similar sentences ignoring other sentences. These sentences will generate the final summary. The disadvantages of the system consist that single-document summarization results are efficient compared to multi-document summarization, the hierarchical algorithm applied is less efficient than other clustering techniques.

3.3 Linear Programming Based Approaches

McDonald was the first who introduce linear programming in automatic document summarization [33]. McDonald's principle is to maximize an objective function satisfying the selection criteria and penalizing the redundancy between the selected

units. The calculation is effected using a set of constraints according to the summary system. Alguliyev et al. (2011) [13], present text summarization approach modeled as a linear programming problem called MCMR. This model tries to optimize relevance, redundancy and summary length. The approach applies to both singles as well as multi-document. The system discovers key phrases in the given document (s), it covers the main content of the input document (s). Also, it reduces redundancy. But models result depends directly on the optimization algorithm. Banerjee et al. (2015) [14], propose an abstractive summarizes begins with identifying the most important document in the multi-document set.

Each sentence in the most important document is initialized in clusters distinct, each sentence in the other documents are assigned to the cluster that has the highest similarity to that sentence. A word graph is generated from the sentences in each cluster; several paths can be extracted from each word graph, in which case they choose the shortest K-paths. To make the most informative and linguistically well-formed sentences, they use an ILP (Integer Linear Programming) based approach, including only the paths that maximize the content of information and linguistic quality of summary.

The ILP system can combine the knowledge of different sentences and present a readable summary; this approach can generate informative summaries by maximizing the selection of content from several sentence clusters, they integrate the linguistic quality and the information to select the coherent sentences in the final summary using this ILP-based approach. The limit in the linguistic variety of the reviews generated is not yet improved; the summary produces incoherent grammatical sentences. Berg-Kirkpatrick et al. (2011) [27], they are learning a model of extract sentences and compression to multi-document summarization.

Their model marks the candidate summaries according to a combined linear model whose features take into consideration the types of n-grams in the summary and the compressions used. Inference in their model can be expressed as ILP and resolved in a reasonable time. The originality of this system that it can use or create reasonable sentences of average length. The Limit is the joint extraction and compression system produce short and less productive sentences. Oliveira et al. (2016) [28], propose an ILP concept-based approach for single document summarization.

Such an approach maximizes the coverage of the important concepts, in summary, avoiding redundancy, and taking into consideration informativeness and readability aspects of the generated summary. The readability of the output summary is improved by incorporating into the ILP model a specific constraint concerning the resolution of dangling co-references and speech analysis. The limit is the inclusion of consistency constraints in ILP models to eliminate co-references and discourse analysis has decreased system performance.

Zhang et al. (2016) [34], propose abstractive cross-lingual summarization framework. The source documents are translated using machine translation system. The approach generates bilingual concepts represented using bilingual elements of source predict, argument structures (PAS) and their target counterparts. In order to maximize the translation quality of (PAS) elements in the final summary, a linear programming algorithm has been applied. The proposed framework can combine relevant information from different original sentences by fusing PAS sharing the same concept.

But the system produces the summary sentence by blending several original sentences and this may violate the correct word ordering.

3.4 Graph Based Approaches

The text is presented using a graph where the vertices are the textual units (concepts or sentences) while the arcs represent the adjacency or semantic relations between nodes. The most popular approaches based graph are: HITS and Google Page Rank. Wan (2008) [15], examines the impact of the document on summarization performance using the document importance and sentence- to-document correlation based on graph ranking process. This approach assumes that the sentences related to an important document and are strongly correlated with this document will be selected in summary.

In order to incorporate the document-level information and the sentence- document relationship, they proposed two-layer links graph models including both sentences and documents. Results show the robustness of the proposed model. The Limit is that The parameters of the proposed model can deteriorate the summary performance. Zhang et al. (2005) [16], generate a new method based on the hub-authority framework. That unites the text content with some cues and explores the subtopics using graph-based sentence ranking algorithm to produce the expected output.

The advantage of the approach is a useful graph-ranking schema in multi-document generic text summarization. Also, top-ranked hub words can be used as keywords to identify document topics in particular applications. The limit of the proposed system is that it's difficult to determine the subtopics. Patil and Brazdil (2007) [17], propose a system called (Summgraph). In this system, the text is presented as a graph where nodes represent the document sentences while the weights on links present the intra-sentence dissimilarity.

This system is focused on the use of the Pfnet concept (PathFinder Network Scaling) to calculate the importance of a sentence in the text. The limitation consists the inability of the model to improve performance. Thakkar et al. (2010) [18], present graph-based methods for text summarization. A score is computed for nodes using graph ranking algorithms: HITS and Page Rank. The algorithm of the shortest path has also been applied to generate the summary.

It is easy to implement, language independent, and it makes an efficient outline by including the most significant parts of the original text. The inconvenient is that Considering the shortest paths for choosing summary sentences may not be enough. Heu et al. (2015) [29], propose "FoDoSu" multi-document text summarization system. Word analysis words frequency and analyzes semantic words importance by exploiting the Flickr tag clusters. The contribution of the word in a document is calculated using the HITS algorithm.

The module of sentence analysis generates the summary by selecting only the highest ranked sentences. The FoDoSu system performs with a low calculation cost during the phase of semantic analysis of the words in the document, FoDoSu effectively selects the sentences containing significant words by exploiting the HITS algorithm with tag clusters during document summaries. FoDoSu analyzes the proper names. The disadvantages consist methods used for semantic analysis of words that are difficult to analyze, such as proper nouns and newly coined words are insufficient.

3.5 Lexical Cohesion Based Approaches

It is primarily based on the cohesion relations between words, such as a lexical chain (LC), a score of a lexical chain of a word (LCS), WordNet (WN), etc. Chen et al. (2005) [19], study the use of lexical chains as a model of several documents written in Chinese to generate an easy and indicative summary. The algorithm of calculating lexical chains using HowNet knowledge database is changed to improve performance and adapt to Chinese compression.

Based on the semantic analysis, the algorithm can eliminate redundant similarities and maintain differences in information content between multi-documents. The approach has excellent performance in capturing meanings and topics of multi-documents, the plan is highly domain- independent, even though its power has illustrated mainly for news-wire texts. The presented system can be used in daily web texts. However the extracted sentences are modified by the algorithm, the summary generated may not be as flexible, concise and coherent.

AL-Khawaldeh and Samawi (2015) [20], develop an LCEAS system in four phases, namely: pre-processing of the text (elimination of stop words, ...), word sense disambiguation: identify a real sense of words. Lexical cohesion based segmentation: distinguish important from non-important sentences in the text. And text-based segmentation for summarization: decide whether the sentence sense is derived from another sentence. The advantage of the system is the elimination of poor sentences in lexical cohesion with word sense disambiguation (WSD), and eliminating of redundant phrases.

The limit is that in the sentence analysis process semantic relationships are insufficient. Azmi and Al-Thanyyan (2012) [21], generate extractive Arabic summary. They create the first summary based on RST (Rhetorical Structure Theory), where they assign a score to each of first summary sentences to produce the final review. The system can let the user define the maximum size of the summary in the form of some words, the percentage of the original or number of sentences although the algorithm failed to generate a review for the document of size 12 and less. Saxena and Saxena (2016) [30], present extractive summary based on the lexical chain approach, wordnet knowledge database is used to identify semantic relations among words, also to create the lexical chains.

These chains will be classified to distinguish the strong ones based on the sense of the lexical chain in a document, the semantic relation between word and the lexical chain, and The utility of each lexical chain to specify its contribution in the report. The selection of sentences included to summary depends on strong lexical chains. The method is significant in extracting relevant or accurate sentences. The Limit is that the processing time which is a little more. The proposed approach takes some time for generating the summary after the complete procedure.

3.6 Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) Based Approaches

This approach uses algebraic theories namely the matrix, the transposition of the matrix, etc. The most used algorithms for constructing a text summarization based

on this approach are LSA (Latent Semantic Analysis), and SDD (Semi-Discrete Decomposition). Wang and Ma (2013) [22], propose LSA-based Text summarization algorithm that Combines term description and sentence description for each topic then select most relevant sentences which include the terms that can best represent the topic. For each subject, several sentences are chosen for the summary, which allows to fully expressing the issue. The limitation consists of the sense of sentences.

Xiong and Luo (2014) [23], propose a novel method for evaluating a subset of sentences based on their capacity to reproduce word projections on singular vectors. The algorithm is computationally efficient. The inconvenient of the system is that it needs another method to produce a better summary. Conroy et al. (2013) [31], present methods for multilingual document summarization.

For term weighting of multilingual single document summarization, three approaches were presented namely: global entropy, the logarithm of frequency, and a personalized variant of TextRank. For multi-document summarization three algorithms were applied: well-known LSA, the more recent latent Dirichlet allocation (LDA) and a new interval non-bounded matrix factorization method (IB- NMF). The evaluation of single document summarization indicates that the approach significantly outperformed the baseline system.

In contrast, the LDA method for term weight was the weak- est of the three and therefore did not improve the performance of the system. Demirci et al. (2017) [32], produce an extractive multi-document summary for Turkish news. The news was collected from various web sources using the JSOUP and RSS Feed frameworks. The collected contents were ranked according to their cosine similarity score as the elements of similar topics.

An LSA based algorithm is applied to identify relations between concepts and sentences then select just the most important. The proposed approach is competitive for short texts. The limits consist some of words observed in the sentences which affect the scoring during LSA. The phrase that has more words considered the most important. The sentence length impacts its importance. Therefore results indicate that performance decreases for long texts.

4 Discussions

Several techniques have been utilized for automatic text summarization task as specified in Table 1. In addition, most of these studies focused on Extractive text summarization, only(Banerjee et al 2015) [14], (Zhang et al. 2016) [34], proposed an abstractive text summarization based on linear programming approach, as well as (Nallapati et al. 2017) [12], generated an abstractive summary using Machine Learning approach.

The comparative study also shows that the majority of proposed systems are monolingual systems in particular in English language, except (Azmi et al. 2012) [21], (Fejer et al. 2014) [26], (AL- Khawaldeh et al. 2015) [20], presented an Arabic text summarization, also one works in Japanese and another in Turkish were covered by (Fukumoto 2004) [5], and(Demirci et al. 2017) [32] respectively. We also noted that there is a considerable lack of multilingual text summarization (Conroy et al. 2013) [31], the only who produced a summary in nine languages, then (Gupta 2012)

Table 1. Recent automatic text summarization.

Authors	Summarization approach	**S**	**M**	**G**	**O**	**E**	**A**	**ML**	**MUL**	**Cross**
Fukumoto 2004 [5]	Statistical	×	×	×		×		× Japanese		
Yong et al 2006 [11]	Machine Learning	×		×		×		× English		
Zhang 2005 [16]	Graph		×	×		×		× English		
Chen et al 2005 [19]	Lexical cohesion		×	×		×		× Chinese		
Li et al 2007 [10]	Machine learning		×		×	×		× English		
Patil et al 2007 [17]	Graph	×		×		×		× English		
Wan 2008 [15]	Graph		×	×				× English		
Ouyang 2009 [6]	Statistical		×		×	×		× English		
Thakkar et al 2010 [18]	Graph	×		×		×		× English		
Agarwal et al 2011 [9]	Machine Learning		×		×	×		× English		
Alguliyev et al 2011 [13]	Linear Programming	×	×	×		×		× English		
Berg-Kirkpatrick et al 2011 [27]	Linear Programming		×	×		×		× English		
Gupta 2012 [7]	Statistical	×		×		×			× Portuguese	
									Brazilian	
									English	
Azmi et al 2012 [21]	Lexical Cohesion	×		×		×		× Arabic		
Wang et al 2013 [22]	LSA	×	×	×		×		× English		
Conroy et al 2013 [31]	LSA, LDA	×	×	×		×			× English	
									Arabic	
									French	
									Czech	
									Greek	
									Hebrew	
									Hindi	
									Spanish	
									Romanian	
									Chinese	
El hadj et al 2013 [24]	Statistical	×	×	×		×			× English	
									Arabic	
Fejer et al 2014 [26]	Machine Learning	×	×	×		×		× Arabic		
×iong et al 2014 [23]	LSA		×	×		×		× English		
Banerjee et al 2015 [14]	Linear Programming		×	×			×	× English		
AL-Khawaldeh et al 2015 [20]	Lexical Cohesion	×		×		×		× Arabic		
Bhatia et al 2015 [25]	Statistical	×		×		×			×	
Heu et al 2015 [29]	Graph		×	×		×		×		
Nallapati et al 2017 [12]	Machine Learning	×		×			×	× English		
Zhang et al 2016 [34]	Linear Programming		×	×			×			×
Saxena and Saxena 2016 [30]	Lexical Cohesion	×		×		×		×		
Oliveira et al 2016 [28]	Linear Programming	×		×		×		×		
Demirci et al 2017 [32]	LSA		×	×		×		× Turkish		

[7], and (El hadj et al. 2013) [24], introduced a multilingual summary included the English language. We also observed that the system of (Conroy et al. 2013) [31] explored the thematic and semantic aspect of text to generate the summary. However, most researchers did not take into count thematic aspect of the text as specified in Table 1. Due to the diversity of performance evaluation metrics of summary: rouge, precision, recall, f-measure. Some works used rouge with its variants [13, 23] while others utilized Precision, Recall and F-measure [32]. In addition these studies used several corpus to evaluate results of the summaries generated: DUC 2002 [23, 22, 17, 15], DUC 2004 [16, 12, 22], DUC 2005 [10, 13, 14], DUC 2006 [10], DUC2007 [10,

10], TSC3 [5], EASC [20]. The comparison of evaluation performance of summary is not significant since the studies did not use the same metrics and corpus.

5 Conclusion

This article presented a survey of text summarization techniques, the investigation has introduced different types of summary. Then a classification of great techniques used in this field has also covered in this paper, with a list of related works that included the limits and advantages of each proposed method in each category. Which is considered a significant contribution to this paper. The research studies have been compared in a tabular form, the purpose of the paper is to help researchers to have a clear view about necessary information of text summarization task, also to choose their appropriate frameworks based on this article.

In the future work, we are going to propose a novel approach for abstractive multi-document multilingual text summarization in particular (Arabic, French and English), with the consideration of the thematic aspect of the text based on additional external semantic resources. We are also going to study the comparison of performance evaluation metrics of text summarization with more details.

References

1. Luhn, H. P.: The Automatic Creation of Literature Abstracts. IBM Journal of Research and Development, vol. 2 , pp. 159–165 (1958)
2. Chaar, N., Sana, L.: Filtrage d'information pour la Construction de Résumés Multi-Documents Guidée par le Profil Utilisateur: le Système REDUIT. Université de Marne-la-Vallée (2004)
3. Khan, A., Salim, N., Kumar, Y. J.: A Framework for Multi-Document Abstractive Summarization Based on Semantic Role Labelling. Applied Soft Computing. Elsevier, vol. 30, pp. 737–747 (2015)
4. Bharti, S. K., Babu, K. S.: Automatic Keyword Extraction for Text Summarization: a Survey (2017)
5. Fukumoto, J., Sugimura, T.: Multi-Document Summarization Using Document Set Type Classification. Proceedings of NTCIR (2004)
6. Ouyang, Y., Li, W., Lu, Q.: An Integrated Multi-Document Summarization Approach Based on Word Hierarchical Representation. Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, Association for Computational Linguistics, pp. 113–116 (2009)
7. Gupta, V., Priya, C., Sohan, G., Anita, B., Shobha, K.: An Statistical Tool for Multi-Document Summarization. International Journal of Scientific and Research Publications, vol. 2 (2012)
8. Gambhir, M., Gupta, V.: Recent Automatic Text Summarization Techniques: a Survey. Artificial Intelligence Review. Springer, vol. 1, pp. 1–66 (2017)
9. Agarwal, N., Gvr, K., Reddy, R. S., Rosé, C. P.: Towards Multi-Document Summarization of Scientific Articles: Making Interesting Comparisons with SciSumm. Proceedings of the Workshop on Automatic Summarization for Different Genres, Media, and Languages, Association for Computational Linguistics, pp. 8–15 (2011)
10. Li, S., Ouyang, Y., Wang, W., Sun, B.: Multi-Document Summarization using Support Vector Regression. Proceedings of DUC, Citeseer (2007)

11. Yong, S. P., Abidin, A. I., Chen, Y. Y.: A Neural-Based Text Summarization System. Transactions on Information and Communication Technologies, WIT Press, vol. 47 (2006)
12. Nallapati, R., Zhai, F., Zhou, B.: SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents, AAAI, pp. 3075–3081 (2017)
13. Alguliyev, R. M., Aliguliyev, R. M., Hajirahimova, M. S., Mehdiyev, C. A.: MCMR: Maximum Coverage and Minimum Redundant Text Summarization Model. Expert Systems with Applications. Elsevier, vol. 38, pp. 14514–14522 (2011)
14. Banerjee, S., Mitra, P., Sugiyama, K., Multi-Document Abstractive Summarization Using ILP Based Multi-Sentence Compression, pp. 1208–1214 (2015)
15. Wan, X.: An Exploration of Document Impact on Graph-Based Multi-Document Summarization. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp. 755–762 (2008)
16. Zhang, J., Sun, L., Zhou, Q.: A Cue-Based Hub-Authority Approach for Multi-Document Text Summarization. Natural Language Processing and Knowledge Engineering, 2005, IEEE NLP-KE'05. Proceedings of 2005 IEEE International Conference on, pp. 642–645 (2005)
17. Patil, K., Brazdil, P.: Text Summarization: Using Centrality in the Pathfinder Network. Int. J. Comput. Sci. Inform. Syst. vol. 2, pp. 18–32 (2007)
18. Thakkar, K. S., Dharaskar, R. V., Chandak, M. B.: Graph-Based Algorithms for Text Summarization. Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference, pp. 516–519 (2010)
19. Chen, Y. M., Wang, X. L., Liu, B. Q.: Multi-Document Summarization Based on Lexical Chains. Machine Learning and Cybernetics, 2005, Proceedings of 2005 International Conference, vol. 3, pp. 1937–1942 (2005)
20. Al-Khawaldeh, F., Samawi, V.: Lexical Cohesion and Entailment Based Segmentation for Arabic Text Summarization (lceas). The World of Computer Science and Information Technology Journal (WSCIT), vol. 5, pp. 51–60 (2015)
21. Azmi, A. M., Al-Thanyyan, S.: A Text Summarizer for Arabic. Computer Speech & Language, vol. 26, pp. 260–273 (2012)
22. Wang, Y., Ma, J.: A Comprehensive Method for Text Summarization Based on Latent Semantic Analysis. Natural Language Processing and Chinese Computing, pp. 394–401 (2013)
23. Xiong, S., Luo, Y.: A New Approach for Multi-Document Summarization Based on Latent Semantic Analysis. Computational Intelligence and Design (ISCID), 2014 Seventh International Symposium. IEEE, vol. 1, pp. 177–180 (2014)
24. El-Haj, M., Rayson, P.: Using a Keyness Metric for Single and Multi Document Summarisation. Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-document Summarization, pp. 64–71 (2013)
25. Bhatia, P., others.: Multilingual Text Summarization with UNL. Computer Engineering and Applications (ICACEA), 2015 International Conference on Advances, pp. 740–745 (2015)
26. Fejer, H. N., Omar, N.: Automatic Arabic Text Summarization using Clustering and Keyphrase Extraction. Information Technology and Multimedia (ICIMU), 2014 International Conference on IEEE, pp. 293–298 (2014)
27. Berg-Kirkpatrick, T., Gillick, D., Klein, D.: Jointly learning to Extract and Compress. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, vol. 1, pp. 481–490 (2011)
28. Oliveira, H. Lima, R., Lins, R. D., Freitas, F., Riss, M., Simske, S. J.: A Concept-Based Integer Linear Programming Approach for Single-Document Summarization. Intelligent Systems (BRACIS), 2016 5th Brazilian Conference. IEEE, pp. 403–408 (2016)

29. Heu, J., Asim, I., Lee, D.: FoDoSu: Multi-Document Summarization Exploiting Semantic Analysis Based on Social Folksonomy. Information Processing and Management. Elsevier, vol. 1, pp. 212–225 (2015)
30. Saxena, S., Saxena, A.: An Efficient Method Based on Lexical Chains for Automatic Text Summarization. International Journal of Computer Applications, Foundation of Computer Science, vol. 144 (2016)
31. Conroy, J., Davis, S.T., Kubina, J., Liu, Y.K., O'leary, D.P., Schlesinger, J.D.: Multilingual Summarization: Dimensionality Reduction and a Step Towards Optimal Term Coverage. Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-Document Summarization, Foundation of Computer Science, pp. 55–63 (2013)
32. Demirci, F., Karabudak, E., İlgen, B.: Multi-document summarization for Turkish news. Artificial Intelligence and Data Processing Symposium (IDAP), 2017 International, pp. 1–5 (2017)
33. McDonald, R.: A Study of Global Inference Algorithms in Multi-Document Summarization. European Conference on Information Retrieval, pp. 557–564 (2007)
34. Zhang, J., Zhou, Y., Zong, C.: Abstractive Cross-Language Summarization via Translation Model Enhanced Predicate Argument Structure Fusing. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), vol. 24, pp. 1842–1853 (2016)

ISSN 1870-4069

Multimodal Neural Machine Translation Using CNN and Transformer Encoder

Hiroki Takushima[1], Akihiro Tamura[2], Takashi Ninomiya[1], Hideki Nakayama[3]

[1] Ehime University, Ehime, Japan

[2] Doshisha University, Kyoto, Japan

[3] The University of Tokyo, Tokyo, Japan

{takushima@ai., ninomiya@}cs.ehime-u.ac.jp
aktamura@mail.doshisha.ac.jp
nakayama@nlab.ci.i.u-tokyo.ac.jp

Abstract. Multimodal machine translation uses images related to source language sentences as inputs to improve translation quality. Pre-existing multimodal neural machine translation (NMT) models that incorporate the visual features of each image region into an encoder for source language sentences or an attention mechanism between an encoder and a decoder cannot catch the relationship between the visual features from each image region. This paper proposes a new multimodal NMT model that encodes an input image using a convolutional neural network (CNN) and a Transformer encoder. In particular, our proposed image encoder extracts visual features from each image region using a CNN then encodes an input image based on the extracted visual features using a Transformer encoder, where the relationship between the visual features from each image region is captured by a self-attention mechanism of the Transformer encoder. Our experiments with English-German translation tasks using the Multi30K data set showed our proposed model improves 0.96 BLEU points against a baseline Transformer NMT model without image inputs and improves 0.47 BLEU points against a baseline multimodal Transformer NMT model without a Transformer encoder for images.

Keywords: Multimodal neural machine translation, machine translation, multimodal learning, neural network, transformer, CNN.

1 Introduction

Various neural network (NN)-based methods have been actively studied in the area of neural machine translation (NMT). Recently, a Transformer NMT model [14] attracted attention for performing state-of-the-art translation above other NMT models. Like (RNN)-based NMT models [13,8] and convolution neural network (CNN)-based NMT models [5], a Transformer NMT model consists of an encoder

that generates intermediate expressions from source language sentences and a decoder that predicts target language sentences from intermediate representations. Each of the Transformer encoder and decoder has a self attention mechanism that catches the relation between the words in a sentence. In particular, the self-attention mechanism in the Transformer encoder catches the relationship between input words in a source language sentence, and the mechanism in the Transformer decoder catches the relationship between output words in a target language sentence. Multimodal NMT [1] uses images related to source language sentences and source language sentences as inputs to improve translation quality.

Multimodal NMT models assume that additional input images could help improve translation performance as clues that decrease translation ambiguity. For example, the English word "bank" has multiple meanings and can be translated into two Japanese words, "銀行 (a business that holds and lends money and provides other financial services)" and "土手 (land along the side of a river or lake)"; therefore, "bank" has translation ambiguity. "Bank" can be translated into "銀行" if a financial image is input, and "bank" can be translated into "土手" if a river image is input. Some of the pre-existing multimodal NMT models incorporate visual features extracted from each region of the input image using a CNN in an attention mechanism between an encoder and decoder of the NMT model; therefore, target language sentences could be generated by input images being reflected through the cross-lingual attention mechanism [3].

In addition, multimodal NMT models that incorporate the visual features of each region into an encoder for source language sentences have been proposed [7,2]. However, pre-existing multimodal NMT models cannot catch the relationship between the visual features from each image region. Therefore, we propose a new multimodal NMT model that encodes an input image using a CNN and a Transformer encoder. Specifically, the transformer decoder in our model generates a target language sentence from the concatenation of the intermediate expression of an input image, which is the output of an encoder for an image (hereinafter referred to as "image encoder"), and that of a source language sentence, which is the output of an encoder for a source language sentence (hereinafter referred to as "source language encoder").

The image encoder first extracts visual features from each image region using a CNN, then encodes an image based on the extracted visual features using a Transformer encoder. Our model could catch the relationship between visual features from each image region by a self-attention mechanism of the transformer encoder in the image encoder. The experiments with the English-German translation tasks using the Multi30K data set [4] showed that our model improved 0.96 BLEU points against a baseline Transformer NMT model that does not use image inputs and improved 0.47 BLEU points against a baseline multimodal Transformer NMT model that does not use a Transformer encoder in the image encoder.

2 Related Work

In this section, we overview the Transformer NMT model [14] we based our model on and describe previous multimodal NMT models.

2.1 Transformer

Figure 1 overviews the Transformer NMT model [14]. The Transformer model accepts a source language sentence $X = (x_1, x_2, ..., x_M)$ as an input and outputs a target language sentence $Y = (y_1, y_2..., y_L)$. In training, objective function $p(Y|X)$ is learned from a parallel corpus. The Transformer model is an encoder-decoder model in which the Transformer encoder generates the intermediate representation h_t $(t = 1, ..., M)$ from the source language sentence X and the Transformer decoder generates a target language sentence Y from the intermediate representation h_t:

$$h_t = \text{TransformerEncoder}(X), \tag{1}$$

$$Y = \text{TransformerDecoder}(h_t). \tag{2}$$

N layers are stacked in the Transformer encoder and the Transformer decoder. Each layer of the encoder is composed of two sublayers, self-attention and point-wise, fully connected layers (hereinafter referred to as "Feed Forward"). Each layer of the decoder is composed of three sublayers, an attention mechanism between the source language and the target language (hereinafter referred to as "cross-lingual attention") and the two sublayers of the encoder. Layer normalization [14] and a residual connection [14] are used between the sublayers in the encoder and decoder. Each attention mechanism $Attention(\cdot)$ (i.e., a self-attention mechanism and a cross-lingual attention mechanism) is computed as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{Q\,K^T}{\sqrt{d_{\text{model}}}}\right)V. \tag{3}$$

where Q, K, and V represent an internal representation of the encoder/decoder and d_{model} is the dimension of the internal representation. The inner product of Q and K represents the similarity between each element of Q and K and is converted to a probability by using the softmax function, which can be treated as weights of attention of Q to K. Finally, the attention mechanism computes a weighted sum of V with the attention weights.

In this way, the attention mechanism generates an expression that reflects the degree of association between a word in Q and that in K. The self-attention mechanism computes the degree of association between words in the same sentence by using the same input source for Q, K, and V. In particular, the self-attention in the encoder catches the degree of association between words in the input sentence by using the internal expressions in the encoder as Q, K, and V.

Meanwhile, the self-attention in the decoder catches the degree of association between words in the output sentence by using the internal expressions in the decoder as Q, K, and V. The cross-lingual attention mechanism computes the degree of association between a word in the source language sentence and that in the target language sentence by using the internal representation of the decoder as Q and the output of the last layer of the encoder as K and V.

Vaswani et al. [14] found that the multi-head attention mechanism that uses multiple attention functions $Attention(\cdot)$ is more beneficial than a single attention mechanism. In the multi-head attention with H heads, Q, K, and V are linearly projected to H

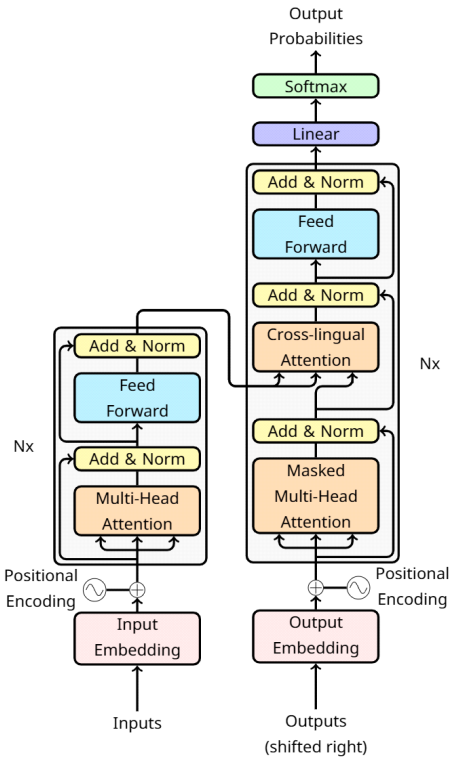


Fig. 1. Transformer NMT Model.

subspaces, and then the attention function is performed in parallel on each subspace. Finally, these are concatenated, and the concatenation is projected to a space with the original dimension. The multi-head attention mechanism $MultiHead(\cdot)$ is represented as follows:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)W^o, \tag{4}$$

$$\text{head}_i = \text{Attention}(QW_i^Q,\ KW_i^K,\ VW_i^V). \tag{5}$$

where $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_k}$ are weight matrices for linear transformations of Q, K, V from d_{model} dimension to $d_k = (d_{\text{model}}/H)$ dimension, respectively, H is the number of head, and Concat is a function that concatenates two matrices. Multi-head attention enables the model to aggregate information from different representation spaces at different positions. The Transformer uses positional encoding (hereinafter referred to as "PE") to encode the positional information of each word in a sentence because the Transformer does not have any recurrent or convolutional structure. PE is calculated as follows:

$$\text{PE}(\text{pos},\ 2i) = \sin(\text{pos}/10000^{2i/d_{\text{model}}}), \tag{6}$$

$$\mathrm{PE}(\mathrm{pos},\ 2i+1) = \cos(\mathrm{pos}/10000^{2i/d_{\mathrm{model}}}). \tag{7}$$

where pos is the absolute position of the word and i is the dimension. At the bottom of the encoder and the decoder in Fig. 1, PE is added to the input embeddings.

2.2 Previous Multimodal NMT

Since NMT appeared, multimodal NMT has been actively researched. For example, many multimodal NMT models have been proposed in the multimodal machine translation shared task [1] at the WMT conference. Calixto et al. [3] improved translation performance by introducing attentions between images and target language sentences into the RNN-based sequence-to-sequence NMT model [8]. Calixto et al. [2] proposed a method for incorporating visual features extracted from input images by using a CNN in the initial hidden state of the RNN encoder and a method for injecting visual features into the input of a decoder.

Huang et al. [7] incorporated both features extracted from each image region by using a CNN and visual features caught by object detection using a region-based convolutional network (R-CNN) [10] in the initial hidden state of the LSTM encoder. Recently, some Transformer-based multimodal NMT models have been proposed. Grönroos et al. [6] added a gating layer to each output of the Transformer encoder and decoder, and their model uses visual features in the gate. They showed that the proposed gating layer in the encoder decreases ambiguity in encoding source language sentences and that in the decoder suppresses the outputs of unnecessary words. Note that such previous multimodal NMT models can not establish the relationship between the visual features from each image region.

3 Proposed Model

In this section, we propose a multimodal NMT model that encodes an input image using a CNN and a Transformer encoder to catch the relationship between the features of each image region. Figure 2 overviews our model.

Our model has two encoders: an image encoder and a source language encoder. In our model, an input image is encoded by the image encoder and a source language sentence is encoded by the source language encoder. Then, the concatenation of intermediate expressions generated by the two encoders is fed into the conventional Transformer decoder and the Transformer decoder generates a target language sentence from the intermediate expression based on image and source language sentence information.

In the rest of this section, we describe the proposed image encoder in Section 3.1. We describe the composition layer that concatenates the intermediate expression of the image encoder and that of source language encoder in Section 3.2. The details of the source language encoder and the decoder of our model are the same as the encoder and decoder of the conventional Transformer NMT model described in Section 2.1, respectively.

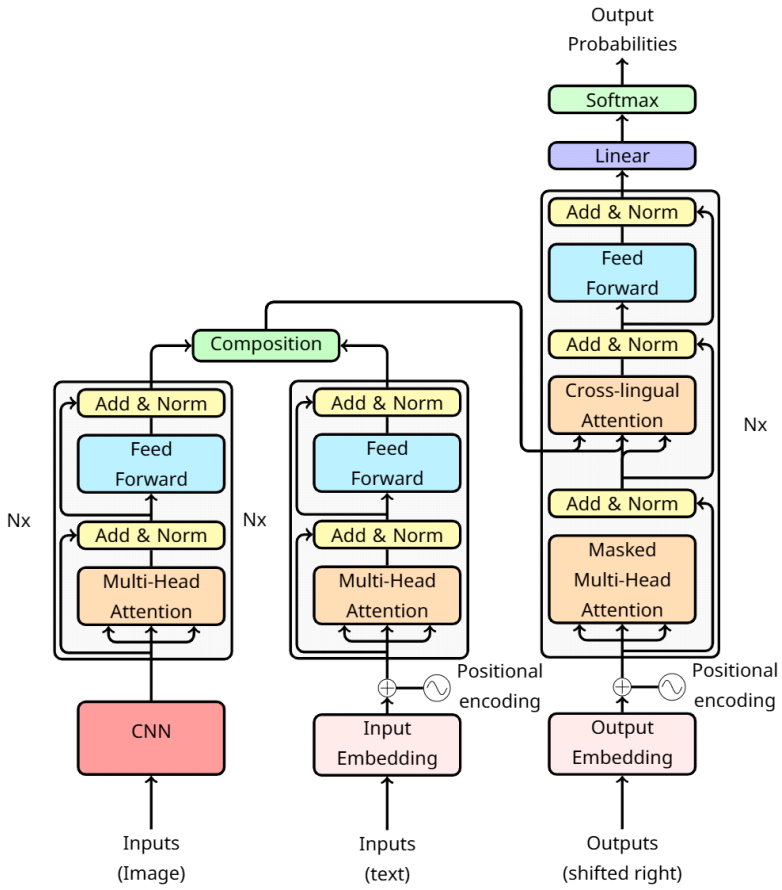


Fig. 2. Proposed Multimodal NMT Model.

3.1 Image Encoder

Our image encoder first uses a CNN to extract visual features from each image region. CNN is a neural network that includes multiple convolution layers and pooling layers. In our experiments, we used VGG16 [12], which has 16 layers of 13 convolution layers and 3 fully connected layers. Then we fed the visual features output by a CNN into the conventional Transformer encoder:

$$\text{feature} = \text{CNN}(\text{image}), \tag{8}$$

$$h_v = \text{TransformerEncoder}(W \cdot \text{feature}). \tag{9}$$

where image is an input image, feature is the visual feature extracted by a CNN, h_v is the intermediate representation generated by the image encoder, and $W \in \mathbb{R}^{d_{\text{feature}} \times d_{\text{model}}}$ is

a weight matrix for linear transformation (from d_{feature} dimension to d_{model} dimension). Note that positional encoding PE is not used in the Transformer encoder of the image encoder.

3.2 Composition Layer

After the intermediate representations of an input image and a source language sentence are generated by the image encoder and the source language encoder, respectively (see Eq. (1) and Eq. (9)), the two representations are concatenated in the composition layer:

$$h = \text{Concat}(h_v,\ h_t). \tag{10}$$

The concatenation h is the output of the proposed encoder and is fed to the decoder of our model.

4 Experiments and Results

This section describes the experiments for evaluating the proposed multimodal NMT model.

4.1 Experiment Setting

We performed the English-German translation tasks using the Multi30K dataset [4]. We prepared 29,000 sentences as a training dataset, 1,014 sentences as a variation dataset, and 1,000 sentences as a test dataset. We used byte pair encoding (BPE) [11] for sub-wording (i.e., words with a low number of occurrences were decomposed into subword units).

The vocabulary set was shared between encoder and decoder, and there were 6,150 words in the vocabulary set. We resized an input image to 256×256, and then used the center cropped 224×224 image as the input of the image encoder. We used the output of the final convolution layer of VGG16 [12] for image regions as the input of the Transformer encoder of the image encoder.

Following the original paper on Transformers [14], the Transformer models had six layers stacked for each encoder and decoder where each layer had 8 heads and the embedded vectors had 512 dimensions ($d_{\text{model}} = 512$). We used Adam for optimization with $\beta_1 = 0.9,\ \beta_2 = 0.98,\ \epsilon = 10^{-9}$. The learning rate was set following the original Transformer paper [14].

The mini batch size was set to 80, and 50 epochs were repeated. We did not perform CNN fine-tuning while training the NMT. Greedy decoding was used to generate target language sentences at inference. BLEU [9] evaluated translation quality. We used the model with the best BLEU score for the validation dataset to evaluate the test dataset.

Table 1. Results.

Method	BLEU
Transformer without image	34.30
CNN+$\text{Trans}_{\text{Dec}}$	34.79
CNN+$\text{Trans}_{\text{Enc}}$+$\text{Trans}_{\text{Dec}}$	35.26

4.2 Evaluation

Table 1 shows the experimental results. In the table, "Transformer without image" means a baseline Transformer NMT model without image inputs. "CNN + $\text{Trans}_{\text{Dec}}$" indicates a baseline multimodal NMT model, where a CNN encoder is used for image encoding and a Transformer is used for translation. Note that "CNN + $\text{Trans}_{\text{Dec}}$" does not use a Transformer encoder for image encoding.

"CNN + $\text{Trans}_{\text{Enc}}$+$\text{Trans}_{\text{Dec}}$" indicates our proposed model, which is comprised of a CNN encoder and a Transformer encoder for image encoding and a Transformer for translation. As shown in the table, our model improved 0.96 BLEU points against the baseline Transformer (Transformer without image), and improved 0.47 BLEU points against the baseline multimodal NMT (CNN + $\text{Trans}_{\text{Dec}}$).

5 Discussion

Table 2 and Fig. 3 show translation examples of each model and the input image for the multimodal NMT models (i.e., the proposed model (CNN+$\text{Trans}_{\text{Enc}}$+$\text{Trans}_{\text{Dec}}$) and the baseline model (CNN+$\text{Trans}_{\text{Dec}}$)), respectively. As Table 2 shows, while a Transformer without an image, which does not use the input image, misses the information of "feuer (fire)," our model and CNN+$\text{Trans}_{\text{Dec}}$, which utilize the input image, successfully include the information of "fire." Moreover, while CNN+$\text{Trans}_{\text{Dec}}$ misses the information of "wand/mauer (wall)", our model successfully includes the information of "wall".

To confirm whether additional input images are effective in our model, we evaluated the last layer's cross-lingual attentions from the word "feuer (fire)" to the input image and those from the word "wand (wall)" to the input image. Figures 4 and 5 visualize the cross-lingual attentions. In Figs. 4 and 5, the clearer region represents a higher attention score. Figures 4 and 5 show that "feuer (fire)" and "wand (wall)" pay attention to the regions representing the words. This indicates that the input images could help prevent missing translations.

To confirm the effectiveness of our model, we evaluated self-attention from a region of a wall to each region of the input image. Figure 6 illustrates the self-attentions. As shown in the figure, a region of a wall is associated with other regions of a wall through self-attentions in the Transformer encoder of the image encoder. The proposed model might avoid missing the word "wand (wall)" by associating multiple regions in the image, while CNN+$\text{Trans}_{\text{Dec}}$ might not capture a "wall" from visual features of individual regions.

Table 2. Translation Example.

Input	man scaling a wall with fire in his hand
Reference	mann erklettert mauer mit feuer in der hand
Transformer without image	ein mann , der eine wand in der hand fordert, geht eine wand in der hand（a man who demands a wall in his hand, goes a wall in his hand）
CNN+$\text{Trans}_{\text{Dec}}$	ein mann , der ein feuer in der hand hält (a man holding a fire in his hand）
CNN+$\text{Trans}_{\text{Enc}}$+$\text{Trans}_{\text{Dec}}$	ein mann geht mit feuer in der hand eine wand entlang（a man is walking along a wall with a fire in his hand）

Fig. 3. Original image.

Fig. 4. Cross-lingual Attention from “fire”.

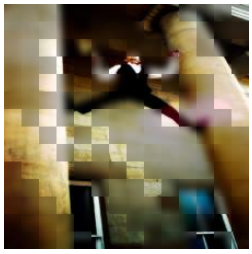


Fig. 5. Cross-lingual Attention from “wall”.

Fig. 6. Self-attention from a region of a wall.

This indicates that our model can determine the associations between regions of an input image, and the associated features can contribute to machine translation.

6 Conclusion

We proposed a new multimodal NMT model that uses a CNN and a Transformer encoder as an image encoder. Our experiments with English-German translation tasks using the Multi30K data set showed that our model outperforms the baseline Transformer NMT model without image inputs and the baseline multimodal NMT model without the Transformer encoder for images.

Through the discussions, we showed that our model can determine the relationship between the visual features from each image region and improve machine translation performance. In the future, we would like to improve our model by incorporating object-detection technology into our image encoder.

Acknowledgments. The research results have been achieved by "Research and Development of Deep Learning Technology for Advanced Multilingual Speech Translation", the Commissioned Research of National Institute of Information and Communications Technology (NICT), Japan. This work was partially supported by JSPS KAKENHI Grant Number JP18K18110, JP25280084.

References

1. Barrault, L., Bougares, F., Specia, L., Lala, C., Elliott, D., Frank, S.: Findings of the third shared task on multimodal machine translation. In: Proceedings of the Third Conference on Machine Translation: Shared Task Papers. pp. 304–323. Association for Computational Linguistics (2018)
2. Calixto, I., Liu, Q.: Incorporating global visual features into attention-based neural machine translation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 992–1003. Association for Computational Linguistics (2017)
3. Calixto, I., Liu, Q., Campbell, N.: Doubly-attentive decoder for multi-modal neural machine translation. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. vol. 1 (2017)
4. Elliott, D., Frank, S., Sima'an, K., Specia, L.: Multi30k: Multilingual english-german image descriptions. In: Proceedings of the 5th Workshop on Vision and Language. pp. 70–74. Association for Computational Linguistics (2016)
5. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.: Convolutional sequence to sequence learning. In: Proceedings of Thirty-fourth International Conference on Machine Learning (ICML2017). pp. 1243–1252 (2017)
6. Grönroos, S. A., Huet, B., Kurimo, M., Laaksonen, J., Merialdo, B., Pham, P., Sjöberg, M., Sulubacak, U., Tiedemann, J., Troncy, R., Vázquez, R.: The MeMAD submission to the WMT18 multimodal translation task. In: Proceedings of the Third Conference on Machine Translation: Shared Task Papers. Association for Computational Linguistics. pp. 603–611 (2018)
7. Huang, P. Y., Liu, F., Shiang, S. R., Oh, J., Dyer, C.: Attention-based multimodal neural machine translation. In: Proceedings of the First Conference on Machine Translation: Shared Task Papers. vol. 2, pp. 639–645. Association for Computational Linguistics (2016)
8. Luong, T., Pham, H., Manning, C. D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics. pp. 1412–1421 (2015)
9. Papineni, K., Roukos, S., Ward, T., Zhu, W.-J.: Bleu: A method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics. pp. 311–318 (2002)
10. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. Curran Associates, Inc., vol. 28, pp. 91–99 (2015)

11. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. vol. 1, pp. 1715–1725 (2016)
12. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Proceedings of 3rd International Conference on Learning Representations, (2014)
13. Sutskever, I., Vinyals, O., Le, Q. V.: Sequence to sequence learning with neural networks. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., Weinberger, K. Q. (eds.) Advances in Neural Information Processing Systems 27. Curran Associates, Inc, vol. 27, pp. 3104–3112 (2014)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems. Curran Associates, Inc., vol. 30 (2017)

ISSN 1870-4069

Query Classification Based on Textual Patterns Mining and Linguistic Features

Mohamed Ettaleb[1], Chiraz Latiri[2], Patrice Bellot[2]

[1] University of Tunis - El Manar,
Faculty of Sciences of Tunis, Tunis,
Tunisia

[2] Aix-Marseille University, Marseille,
France

mouhamed.taleb@hotmail.fr, chiraz.latiri3@gmail.com,
patrice.bellot@lis-lab.fr

Abstract. We argue that verbose natural language queries used for software retrieval contain many terms that follow specific discourse rules, yet hinder retrieval. Through verbose queries, users can express complex or highly specific information needs. However, it is difficult for search engine to deal with this type of queries. Moreover, the emergence of social medias allows users to get opinions, suggestions, or recommendations from other users about complex information needs. In order to increase the understanding of user needs, a tasks, as the CLEF Social Book Search Classification Track, the aims is to investigates how systems can automatically identify book search requests in online forums. In this respect, we introduce in the present paper a new approach to automatically detect the type of each thread. Our proposal aims to identify book search queries by syntactic patterns, association rules between terms and textual sequences mining.

Keywords: Classification, association rules, syntactic patterns, sequences mining.

1 Introduction

The exponential growth of social medias encourage the use of collective intelligence in a spirit of online collaboration. Through these social medias, users can get opinions, suggestions, or recommendations from other members. It is often difficult for users to express their information needs in a search engine query [4]. These social medias allow users to express complex or highly specific information needs through verbose queries.

In this context, this type of queries provides detailed information which can not be found in user profile as the expectations of the users and their tastes. Verbose queries, expressed in natural language, are characterized by their length and a complex structure which provide much more context for a more accurate understanding of users needs. Thereby, it is important to find methods that allow to deal effectively with such type of needs.

The intrinsic characteristics of this type of query allow us to highlight the use of processes from the NLP to infer semantic aspects and thus potentially better interpret users needs. The Social Book Search Lab on the CLEF 2016 conference consisted of three tracks: suggestion, mining and interactive track. Within this work we describe our approach on the mining track which is a new one in SBS 2016 edition and investigates two tasks:

a) Classification task: how Information Retrieval Systems can automatically identify book search requests in online forums,
b) Linking task: how to detect and link books mentioned in online book discussions.

Our contribution deals only with the classification task. The final objective of this task is to identify which threads on online forums are book search requests. Thereby, given a forum thread with one or more posts, the system should determine whether the opening post contains a request for book suggestions (*i.e.*, binary classification of opening posts). In this respect, we propose to use three types of approaches, namely, an approach based on association rules between terms, textual sequences mining, and an NLP method which relies on nouns, verbs and syntactic patterns extraction (*i.e.*, compound nouns), to improve the classification efficiency.

Then, we use the Naive Bayes classifier with WEKA to specify the user's intentions in the requests. In experimental validation, we focus on the analysis of specific needs which are books recommendation but it is worth noting that this approach could be generalized. We believe that specific preprocessing can be set up depending on the nature of the need and the subject of search. The limitations of traditional approaches are overcome, in order to achieve a better representation of user needs.

To our knowledge no study has focused on pattern mining combined with NLP technique in this task as source of the query classification. The paper is structured as follows: related work on classification system is presented in Section 2. Then, a detailed description of our approach is presented in Section 3. Section 4 focuses on an application framework. Section 5 presents the experiments and the results. The Conclusion section wraps up the article and outlines future work.

2 Query Classification Systems

While considering web search, building a classification system that can automatically classify a large portion of the general query stream with a very good level of accuracy is highly challenging. Web traffic at major search engines often reaches hundreds of millions of queries per day, and web queries are topically ambiguous and very short [10]. Classifying these queries into categories is both a difficult and an important task. So, this problem complicates manual categorization of queries (for studying, or to providing training data) poses great challenges for automated query classification [8].

Many studies in query classification classify queries into many different categories based on the information needed by the user. In [5] considered three different categories of queries, namely: *informational queries*, *navigational queries* and *transactional queries*. The majority of works focus on the type of the queries, rather than topical

classification of the queries. Some works are dealing with the problem of classifying queries into a more complex taxonomy containing different topics. To classify queries based on their meaning, some works considered only information available in queries (*e.g.*, in [2] authors only used terms in queries). Some other works attempted to enrich queries with information from external online datasets, *e.g.*, web pages [5, 16] and web directories [17]. The context of a given query can provide useful information to determine its categories. Previous studies have confirmed the importance of search context in query classification.

In [3], authors considered a query as a distribution of terms over topics and mapped into high dimensional space for sparse representation, they used LDA topic model for latent topic extraction and word distribution over topics. The Objective of the work is to classify the query into a topic class label by considering the query keywords distributed over various topics. In [15], authors applied feature expansion using word embedding based on word2vec. They applied this approach using GoogleNews and IndoNews data set on Naive Bayes, SVM, and Logistic Regression classifiers.

In [12], authors proposed a method which takes advantage of the linguistic information encoded in hierarchical parse trees for query understanding. This was done by extracting a set of syntactic structural features and semantic dependency features from query parse trees. In [7], authors proposed a semi-supervised approach for classifying microblogs into six classes. They initially train a classifier with manually labeled data to probabilistically predict the classes for a large number of unlabeled tweets, then they train a new classifier also using the probabilistically predicted labels for the above mentioned unlabeled tweets, and iterate the process to convergence.

The approach introduced in [6] performs query expansion on all query terms upon distinguishing short and verbose queries with promising results. The aims of this method was to show that automatic query classification by verbosity and length is feasible and may improve retrieval effectiveness. Notice that our framework for classifying book search requests in online forums is based on verbose queries, the queries have to be much longer than the conventional ones in a freer style and include a large amount of descriptions of the user's interests.

3 Proposed Approach

The use of verbose queries that have characteristics such as to be long, infers some difficulties to understand user information needs and to interpret them. To overcome this hamper, a subset selection of the original query (*i.e.*, a "sub-query") is proposed for improving these queries representation.The impact of the different types of relations that can exist between original query words is focused. For example, some query words may form a compound nouns such as "suggestion book" and others may describe frequent sequences of terms such as "suggest me please". In this section, three methods for book search request classification are proposed. The first one is based on the sequence mining technique to extract frequent sequences [1] from textual content requests, while the second one consists in extracting linguistic features as verbs, nouns and compound nouns. The last one focused on the association rules to extract useful knowledge and correlations between terms from the queries.

Table 1. Example of frequent sequences extracted from SBS collection.

Most frequent sequences	Support
i_need	18
any_suggestion	13
suggest_me_please	7
any_recommendation_Book	9
any_recommendation_Book_about	5

3.1 Textual Sequence Mining

In order to generate textual frequent sequences features, we propose to extract the frequent and contiguous subsequences in a sequences database and the subsequences whose occurrence frequency is no less than minimum support threshold *minsupp*. In our case, a subsequence is ordered sequence of variable size of k words occurring in the query. We are looking for patterns that are necessarily contiguous, because, if we want to take non−contiguous (gappy) patterns into account, the number of features increases exponentially with the size of the text.

Furthermore, most of these patterns will be mere noisy. To overcome both issues, sequential pattern mining can be used to efficiently extract a smaller number of the most frequent features. To address book search requests classification in an efficient and effective manner, we claim that a synergy with some advanced text mining methods in verbose queries, especially sequence mining [1], is particularly appropriate.

However, extracting the frequent sequences of words on verbose queries helps to select good features and improve classification accuracy, mostly because of the huge number of potentially interesting frequent sequences that can be drawn from a verbose queries collection (Table 1). LCM_seq[3][13] is a variation of LCM[4][18] for sequence mining. It generates all frequently appearing sequence patterns from given database, each transaction is considered to be a sequence. The algorithm follows the scheme so called prefix span, but the data structures and processing method are LCM[18] based.

Definition 1 sequence $S = \langle t_1, ..., t_j, ..., t_n \rangle$ such that $t_k \in \tau$ and n is its length, is a $n - termset$ for which the position of each term in the sentence is maintained. S is called a $n - sequence$.

3.2 Linguistic Features

Textual features extraction is the process of transforming what is essentially a bag of words into a feature set that is usable by a classifier. For the features extraction, two steps are proposed, namely: (1), TREETAGGER[5] was used for annotating text with part-of-speech and lemma information. We noticed that the bag of words model is the

[3] http://research.nii.ac.jp/∼uno/code/lcm_seq.html
[4] LCM : Linear time Closed itemset Miner
[5] http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/

classical method. It doesn't take into account the order of the words, or how many times a word occurs; (2), only the linguistic features are extracted, so we focus on terms that are either verbs, common nouns and proper nouns. The rationale for this focus is that nouns are the most informative grammatical categories and are most likely to represent the textual content. This selection helps the system to identify the category of queries and enhance accuracy.

Finally, we propose to keep the dependencies and relationships between words through language phenomena. The tagged corpus is used to extract a set of compound nouns by the identification of syntactic patterns as detailed in [9]. The pattern is a syntactic rule on the order of concatenation of grammatical categories which form a noun phrase. 12 syntactic patterns to extract the compound nouns are defined namely:

- Four syntactic patterns of size two: Noun-Noun, Adjective-Noun, Noun-Verb and Verb-Noun.
- Six syntactic patterns of size three: Adjective-Noun-Noun, Adjective-Noun-Gerundive, Noun-Verb-Adjective, Verb-Adjective-Noun, Noun-Noun-Verb and Adjective-Noun-Verb.
- Two syntactic patterns of size four: Noun-Verb-Adjective-Noun and Noun-Noun-Adjective-Noun.

Then, compound nouns are extracted by the identification of syntactic patterns. For example, the compound nouns *suggestion fantasy* and *looking new book* are extracted based on the syntactic patterns respectively Noun-Noun and Verb-Adjective-Noun. The purpose is to extract semantically meaningful unit of text from a query. In the particular task of mining track for classification, units that contain *needs books* are captured. This type of patterns is important because it expresses and contain words that convey opinions about aspects of entities.

3.3 Statistical Approach based on Association Rules Inter-terms

The main idea of this approach is to extract a set of non redundant rules, representing inter-terms correlations in a contextual manner. We select these rules that convey the most interesting correlations amongst terms, to help the classifier learn to detect the type of queries. We generate the association rules using an efficient algorithm for mining all the closed frequent termsets.

Definition 2 An association rule, *i.e.*, between terms, is an implication of the form $R : T_1 \Rightarrow T_2$, where T_1 and T_2 are subsets of τ, where $\tau := t_1...t_n$ is a finite set of n distinct terms in the collection and $T_1 \cap T_2 = \theta$. The termsets T_1 and T_2 are, respectively, called the *premise* and the *conclusion* of R. The rule R is said to be based on the termset T equal to $T_1 \cup T_2$. The *support* of a rule $R : T_1 \Rightarrow T_2$ is then defined as:

$$Supp(R) = Supp(T). \tag{1}$$

while its *confidence* is computed as:

$$Conf(R) = Supp(T) \div Supp(T1). \tag{2}$$

```
<?xml version="1.0" encoding="UTF-8"?>
<thread id="2/2562">
 <title>Nobel Prize Literature</title>
 <group id="southamericanfictio">South American Fiction-Argentine Writers</group>
 <posts>
   <post id="1">
     <username>carmenDC</username>
     <timestamp>Oct 5, 2006, 6:01pm</timestamp>
     <raw_text>Hello: It's been a while since a Latin American writer won the nobel
     for Literature. Octavio Paz was our last Nobel in 1990.
     Any suggestions on who should be nominated?</raw_text>
     <authors_mentioned/>
     <works_mentioned/>
   </post>
 </posts>
</thread>
```

Fig. 1. Example of LT data format.

An association rule R is said to be valid if its confidence value, *i.e.*, $Conf(R)$, is greater than or equal to a user-defined threshold denoted $minconf$. This confidence threshold is used to exclude non valid rules.

3.4 Mining and Learning Process

The thread classification system serves to identify which threads on online forums are book search requests. Our proposed approach based on text mining is depicted in **Processus 1**. The classification threads process is performed on the following steps: Firstly, annotating the selected threads with part−of−speech and lemma information using TreeTagger. The extracted attributes like nouns, verbs and Nominal syntagms from the annotated threads are denoted by $Feat1$.

Secondly, the sequence of terms and the association rules inter-terms are generated using respectively the efficient algorithms **LCM_seq** and **CHARM**. The matrix $M1$(threads_attributes) was built by multiplying five times the positive instances to balance between positive and negative instances. Then, the classification model using the naive Bayesian classifier[6] was generated. Finally, the prediction classification results C_{ti} are Extracted for test requests.

4 Experimental Framework

To evaluate our approach, the data provided by CLEF SBS Mining track track 2016[7] are used. This task focuses on detecting and linking book titles in online book discussion forums, as well as detecting book search requests in forum posts for automatic book recommendation.

This text mining track is composed of two tasks: classification and linking. The classification task consists to identify book search requests in online forums. The

[6] The Bayesian Classification represents a supervised learning method as well as a statistical method for classification

[7] http://social-book-search.humanities.uva.nl/\#/data/mining

Algorithm 1 Book search requests classification

Requires: List of recommendation queries file

1: $Q_{Train} \leftarrow < q_1, ..., q_{n-1}, q_n >$
2: $Q_{Test} \leftarrow < q_{T1}, ..., q_{Tn-1}, q_{Tn} >$
3: $Feat_1 \leftarrow$ extract the most frequent sequences of terms and the association rules inter-terms from Q_{Train}
4: Tokenize Q_{Train}
5: Remove stop words from Q_{Train} and Q_{Test}
6: **for each** $q_i \in Q_{Train}$ **do**
7: $Feat_2 \leftarrow$ select syntactic patterns and linguistic features
8: $M_1 \leftarrow Q_{Train}-$features matrix
9: Train the classifier based on these features(*Feat1* and *Feat2*).
10: $M_2 \leftarrow Q_{Test}-$features matrix
11: **for each** $q_{ti} \in Q_{Test}$ **do**
12: Run the classification model.
13: $C_{ti} \leftarrow$ predict class for $Qtest_i$

linking task consists to detect and link books mentioned in online book discussions. Concerning the classification task two data sets are used: LibraryThing[8] (LT) and Reddit[9]. These data sets provide discussion threads around several subjects including the recommendation of books provided by users. Note that the linking task use only a data set extracted from LT.

4.1 Reddit

The training data contains threads from the *suggestmeabook subreddit*[10] as positive examples and threads from the books *subreddit* as negative examples. A *subreddit* corresponds to a sub-section of the site devoted to a specific theme, for example, *suggestmeabook subreddit* is a specific sub-section dedicated to book recommendation.

In this *subreddit*, it is possible to find queries of this type: "*I'd like to start reading Joyce, but I feel like I need something to break me in to his style. Suggestions?*". On average, the written posts are composed of 5 sentences and 90 words. The Reddit data consists on 248 labelled threads for training, and 89 labelled threads for testing. These posts are expressed in natural language by users.

4.2 LibraryThing

LibraryThing is concerning the data set provided for the linking task, 200 threads (3619 posts) labelled with touchstones (links to unique book IDs, on the post level) for the training are available. In the test, 5097 book titles are identified in 2117 posts. For the classification task, 2000 labelled threads for training and 2000 labelled threads for testing are available. These posts are composed of: the date the post was written, the

[8] https://www.librarything.com/
[9] https://www.reddit.com/
[10] https://www.reddit.com/r/suggestmeabook

content of the post, the post id, the user name and the thread id. For both tasks, the same type of queries is used, an example of book recommendation query: "*I am looking for a non-fiction history book on the background to and the actual time of the Boer War in South Africa. Can anyone suggest any good titles?*". On average, the written posts are composed of 4 sentences and 77 words.

5 Experiments and Results

5.1 Classification Task Experiments

For the evaluation, official measure provides by CLEF SBS 2016, which is accuracy, are used. Four thread sets are used: LibraryThing−TRAIN and Reddit−TRAIN are used for training (TRAIN), LibraryThing−TEST and Reddit−TEST are used for testing (TEST). Then, the feature set of frequent sequences is extracted using LCM_SEQ[11] which is a variation of LCM[12] for sequences mining. LCM is commonly applied to discover frequently appearing patterns.

We propose to keep all sequence of words of variable size between 2 and 5. Next, we used CHARM to select all association rules inter-term. As parameters, CHARM takes minsupp = 5 as the relative minimal support and minconf = 0.7 as the minimum confidence of the association rules [11]. While considering the $Zipf$ distribution of each collection, the minimum threshold of the support $minsupp$ values was experimentally set in order to eliminate marginal terms which occur in fewer documents, and therefore they are not statistically important when occurring in a rule.

Then, the collection of texts composed of n threads and m features are taken and represents it as an $n \times m$ thread-feature matrix. The elements M_{ij} of the thread-feature matrix are binary and indicate whether a feature exists in the thread or not.

$$M_{i,j} = \begin{cases} 1 & \text{if the feature exists in the thread,} \\ 0 & \text{else.} \end{cases} \tag{3}$$

Then, using WEKA[13], we analyze the accuracy level (error rate) and compare the performance of the classifiers when using the default setting provided by WEKA.

After the training, the test threads are selected. Then, the learned classification model is applied to predict the category of threads, for each of the threads in the test set, the aim is to detect if the opening post of this thread is a book search request or not.

We used three learning algorithms, this choice being explained by the fact that they often showed their effectiveness in text analysis tasks: SVM [19], J48 [14] and Naive Bayes [20]. Finally, we applied a 10-fold cross validation. In the test sets, eight runs are conducted according to the approaches described in Section 3: four runs on the LIBRARYTHING data collection and four runs on the REDDIT data collection.

For each run, we propose to combine different set of features and we used the Naive Bayes classifier that we gave the best scores in the training set

[11] http://research.nii.ac.jp/∼uno/code/lcm_seq.html
[12] LCM : Linear time Closed itemset Miner
[13] http://www.cs.waikato.ac.nz/ml/weka/

(see table2). The following runs are conducted to derive the classification model using Naive Bayesian classifier:

Table 2. Comparing classifier algorithms in terms of Precision and Accuracy.

	Naive Bayes classifier		**J48 classifier**		**SVM classifier**	
Detailed Accuracy	Precision	*Accuracy*	Precision	*Accuracy*	Precision	*Accuracy*
Run-NVC	0.844	0.852	0.637	0.639	0.79	0.79
Run-Seq	0.820	0.832	0.630	0.634	0.763	0.771
Run-ART	0.839	0.845	0.626	0.634	0.803	0.810
Run-All-features	0.859	0.869	0.642	0.650	0.812	0.818

- **Run-LF(NVC)**: the features Nouns, Verbs and Compound nouns are combined.
- **Run-Seq**: the features of the sequence of words have extracted using LCM_seq algorithm taking $minsupp = 5$ as parameters.
- **Run-ART**: We applied the CHARM algorithm to select the set of association rules inter-terms, this latter is combined with the *Run-LF* to extract the classification model.
- **Run-All-features:** the features Compound nouns are combined with the sequence of words using the same parameters as *Run-LF(NV).Seq*.

5.2 Results

The results obtained by our approach on the classification task requests are evaluated in two metrics, which are the $Accuracy$ and $Precision$. It simply measures how often the classifier makes the correct prediction. Where: $Accuracy$ is the ratio between the number of correct predictions and the total number of predictions (the number of test data points), and $Precision$ is the proportion of correct positive classifications from cases that are predicted as positive.

$$\textbf{Accuracy} = (TP + TN) \div (TP + TN + FP + FN), \tag{4}$$

$$\textbf{Precision} = (TP) \div (TP + FP). \tag{5}$$

where: TP is the number of True positives, FP is the number of False positives, TN is the number of True Negatives and FN is the number of False Negative. Table 3 summarizes classification task results conducted on the LIBRARYTHING collection. These results highlight that the combination of Bag of words features (*i.e.* nouns and verbs) and compound nouns performs the best in term of accuracy, (*i.e.*, $Run - LF(NVC)$).

We note also that the combination of NLP techniques with patterns mining, (*i.e.*, $Run-All-features$) increases accuracy compared to the use of only NLP techniques. It's noted also that, the runs give very similar results and therefore the differences are

Table 3. Classification of the LibraryThing Threads.

Runs	Accuracy	Precesion
Run-LF(NVC)	90.98	83.04
Run-Seq	90.24	82,63
Run-ART	90.35	82.75
Run-All-features	**91.13**	**83.18**

Table 4. Classification of the Reddit posts.

Runs	Accuracy	Precesion
Run-LF(NVC)	81.92	74,23
Run-Seq	82.02	74.41
Run-ART	81.75	74.09
Run-All-features	**82.23**	**75.03**

Table 5. Comparison results of the Librarything posts.

Runs	Accuracy
Our Best-Run	91.13
Official Run	**94.17**
Medium Run	90.83
Worst Run	74.82

Table 6. Comparison results of the Reddit posts.

Runs	Accuracy
Our Best-Run	**82.23**
Official Run	82.23
Medium Run	78.65
Worst Run	74.16

not seem really significant. Table 4 describes classification task results conducted on the REDDIT collection.

We notice also that when we used the sequences of words, the results are well-performed and increase accuracy (*i.e,* $Run - Seq$. However, we noticed that the association rules between terms (*i.e,* $Run - ART$) worked well in the classification of queries, this is justified by the fact that the association rules allowed us to find the terms having a strong correlation with the query's terms.

The highest accuracy are obtained in the two collections when we combined all the features(*i.e.*, $Run - All - features$). Finally, all these experiments clearly show that the proposed approach allows to detect significantly the classes of queries. These improvements show the interest of combining the features selection to learning models.

Table 5 and 6 present the comparative results with all participants[14]. Our run is best-performed in the evaluation on Reddit collection, and our best results on the LibraryThing collection are ranked sixth in term of accuracy.

The best run on Reddit collection is performed with the sequences of words and the verbs as features for classification. This result confirms that mining sequences is useful for classification task. It's worth noting that the obtained results shed light that our proposed approaches, based on text mining and NLP techniques, offer interesting results and helps to identify book search requests in online forums.

6 Conclusions

The originality of the proposed approach deals with verbose queries of book recommendation in order to identify the type of request and the associated need. We presented our approach for the 2016 Social Book Search Track, especially for the SBS Mining track. In the different runs dedicated for book search requests classification, we tested four approaches for features selection, namely : Bag of linguistic features (*i.e.*, nouns and verbs), compound nouns, sequences mining and association rules between terms, and their combination.

We showed that combining Bag of linguistic features (*i.e.*, nouns and verbs) and compound nouns improves accuracy, and integrating sequences and association rules in classification process enhances the performance. So, the results confirmed that the synergy between the NLP techniques (textual patterns mining and nouns phrases extraction) and the classification system is fruitful.

References

1. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the Eleventh International Conference on Data Engineering. pp. 3–14 (1995)
2. Beitzel, S. M., Jensen, E. C., Frieder, O., Grossman, D. A., Lewis, D. D., Chowdhury, A., Kolcz, A.: Automatic web query classification using labeled and unlabeled training data. In: 2005: Proceedings of the 28th Annual International. Conference on Research and Development in Information Retrieval. pp. 581–582 (2005)
3. Bhattacharya, I., Sil, J.: Query classification using lda topic model and sparse representation based classifier. In: Proceedings of the 3rd IKDD Conference on Data Science. pp. 1–2 (2016)
4. Biskri, I., Rompre, L.: Using associated rules for query reformulation. In: Next Generation Search Engine: Advanced Models for Information Retrieval. IGI_Global, pp. 291–303 (2012)
5. Broder, A. Z., Fontoura, M., Gabrilovich, E., Joshi, A., Josifovski, V., Zhang, T.: Robust classification of rare queries using web knowledge. In: SIGIR 2007: Proceedings of the 30th Annual International Conference on Research and Development in Information Retrieval. pp. 231–238 (2007)
6. Buccio, E. D., Melucci, M., Moro, F.: Detecting verbose queries and improving information retrieval. Inf. Process. Manage., vol. 50, no. 2, pp. 342–360 (2014). doi: 10.1016/j.ipm.2013.09.003

[14] http://social-book-search.humanities.uva.nl/#/mining16

7. Chen, Y., Li, Z., Nie, L., Hu, X., Wang, X., Chua, T., Zhang, X.: A semi-supervised bayesian network model for microblog topic classification. In: COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers. pp. 561–576 (2012)
8. Gravano, L., Hatzivassiloglou, V., Lichtenstein, R.: Categorizing web queries according to geographical locality. In: Proceedings of the 2003 International Conference on Information and Knowledge Management. CIKM. pp. 325–333 (2003)
9. Haddad, H.: French noun phrase indexing and mining for an information retrieval system. In: String Processing and Information Retrieval, 10th International Symposium. pp. 277–286 (2003)
10. Jansen, B. J., Spink, A., Saracevic, T.: Real life, real users, and real needs: a study and analysis of user queries on the web. Inf. Process. Manage., vol. 36, no. 2, pp. 207–227 (2000)
11. Latiri, C. C., Haddad, H., Hamrouni, T.: Towards an effective automatic query expansion process using an association rule mining approach. J. Intell. Inf. Syst., vol. 39, no. 1, pp. 209–247 (2012)
12. Liu, J., Pasupat, P., Wang, Y., Cyphers, S., Glass, J. R.: Query understanding enhanced by hierarchical parsing structures. In: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding. pp. 72–77 (2013)
13. Nakahara, T., Uno, T., Yada, K.: Extracting promising sequential patterns from RFID data using the LCM sequence. Knowledge-Based and Intelligent Information and Engineering Systems - 14th International Conference, pp. 244–253 (2010)
14. Quinlan, J. R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
15. Setiawan, E. B., Widyantoro, D. H., Surendro, K.: Feature expansion using word embedding for tweet topic classification. 2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA), pp. 1–5 (2016)
16. Shen, D., Pan, R., Sun, J., Pan, J. J., Wu, K., Yin, J., Yang, Q.: Query enrichment for web-query classification. ACM Trans. Inf. Syst., vol. 24, no. 3, pp. 320–352 (2006)
17. Shen, D., Sun, J., Yang, Q., Chen, Z.: Building bridges for web query classification. In: SIGIR 2006: Proceedings of the 29th Annual International Conference on Research and Development in Information Retrieval. pp. 131–138 (2006)
18. Uno, T., Kiyomi, M., Arimura, H.: LCM ver.3: Collaboration of array, bitmap and prefix tree for frequent itemset mining. Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations, pp. 77–86 (2005)
19. Vosecky, J., Leung, K. W. T., Ng, W.: Searching for quality microblog posts: Filtering and ranking based on content analysis and implicit links. Springer Berlin Heidelberg. OSDM'05, ACM. Nw York, NY, USA, pp. 397–413 (2012)
20. Yuan, Q., Cong, G., Thalmann, N. M.: Enhancing naive bayes with various smoothing methods for short text classification. Proceedings of the 21st International Conference on World Wide Web. WWW '12 Companion, ACM, New York, NY, USA, pp. 645–646 (2012). doi: 10.1145/2187980.2188169

ISSN 1870-4069

Restructuring Spoken Corpus for Streaming Emulation

Jan Oldřich Krůza

Charles University,
Institute of Formal and Applied Linguistics,
Faculty of Mathematics and Physics,
Czech Republic

kruza@ufal.mff.cuni.cz

Abstract. This paper shares the experience of transforming the way of storing a spoken corpus into more numerous smaller files. Discussed is the motivation: a change in the usage of the data; the corpus itself: a 1000-hour set of recordings of a single speaker; and decisions to be made for the restructuring, reasons for them and their impact.

Keywords: Speech recognition, cassette digitization, recording on magnetic tapes.

1 Introduction

The way a volume of data is stored impacts all subsequent processing and exploitation [7]. A decision on storing the data has to be made as soon as the data is being acquired. Obviously, the difficulty of changing the way of storing the data increases with the volume of the data stored and with more parties relying on it. The relying parties can be in-house software tools or even public interfaces (I hesitate to say API because we're talking about access to data, not to applications). In my case, the data at hand is a collection of speech recordings of total magnitude of about one thousand hours. The decisions to be made span many levels and the highest levels dictate further questions.

Will the data be stored as files on a traditional file system or in a database or in a yet another way? Will the whole data set be contained on one physical storage device or will it be distributed over several? How will the audio material be divided into individual files and directories? What additional metadata will be maintained [1]? In the following chapters, I shall list the original structuring for the Spoken corpus of Karel Makoň, provide reasons that led to that structuring, explain why a change was necessary, and describe the journey to realizing it.

2 The Spoken Corpus of Karel Makoň

The corpus arose as a set of amateur recordings on magnetic tapes. The author of the talks, Mr. Karel Makoň [2], a Czech mystic, started giving talks in private groups after

his deportation into a concentration camp in 1939, and stopped in 1991, two years before his death. His regular listeners were recording his talks. I don't know exactly when the recording started but the earliest denoted year is 1973. The earliest year I could make out from analyzing the contents is 1970 on an undated recording. Makoň had groups of attendants in various places of then Czechoslovakia.

There was a group in Pilsen, a group in Prague and a group in Gottwaldov, today's Zlín. Some of the recordings are from what we would now call retreats – meetings in a group in detached places that lasted several days. One such place was in Kaly near Brno in south Moravia. Another was in the cottage Čeřínek in the Bohemian-Moravian Highlands. Yet another was in Žiar, Slovakia. Other recordings are mostly from meetings hosted by one of the members.

2.1 Corpus Content

The talks can be seen as accompaniment to the author's written opus. Karel Makoň wrote twenty seven books on his own and translated and commented twenty eight books of other authors. Since the culmination of his spiritual path in the concentration camp in Sachsenhausen, where he acquired profound understanding of symbolism in Christianism and other traditions, he kept translating his experience to words in order to enable others to find enlightenment. His work can be compared to that of other modern spiritual masters, though there are pecularities.

Makoň bases his teachings mainly on Christian mystic, drawing heavily from the catholic tradition. He repeatedly uses the parable of the prodigal son and the parable of the talents in his arguments. He stresses that the words of Jesus must always be considered within the context of his deeds and always as a whole. He refers very often to Saint Teresa of Ávila, Augustine of Hippo and Padre Pio among others. He criticizes the catholic tradition for some points that he claims form an obstacle on the path to God and are in conflict with Jesus' intent.

One such example is the belief that the eternal life can only be reached after the physical death and that a life of virtues leads to it. Makoň claims that the eternal life – being consciously eternal – can only be reached within the physical body and that virtues alone never suffice to this end. He also draws from ancient Indian tradition, notably referring to Sat-Chit-Ananda as the three qualities of God. He translated and commented a book by Sri Aurobindo Ghos and draws from it. He also often refers to the life of Siddhartha Gautama Buddha, notably for his enlightenment in the moment of discarding the path of asceticism.

He suggests that the traditional hinduistic concept of re-incarnation should be revised firstly because a better understanding of what happens with the soul is at hand, secondly because, as he claims, correctly used Christianity leads to redemption within one incarnation. Aside from the symbolism in the New Testament, Makoň often uses milestones in his own life for explaining the general spiritual path. The most important of these milestones are repeated surgery without anaesthesia in the age of six, and confrontation with his own certain death in the concentration camp.

2.2 Original Recording

The vast majority of the recordings have been taken by one of the members of the Gottwaldov group. Fortunately, these recordings were taken using the best amateur technology available at that time and stored very carefully. Recordings from other places also exist but rather as rarities, with inferior acoustic quality and lacking systematic identification. A part of the recordings (30% of total length) was taken onto reels, the rest onto cassettes. Mostly, there was an identifier for a reel tape or for a cassette. Some pieces were unlabeled and some shared a label.

The majority of the recordings are cassettes with identifiers of the format `YY-NN`, for example `85-05` is the fifth recording taken in the year 1985. These occupy 686 resulting files out of total 802 files originating in cassettes. Of the 39 self-digitized reel-to-reel tapes, 36 were recorded at 9.53 [cm per second]. The remaining three in 2.38 [cm per second]. The latter took six hours to play back in one direction and the quality was heavily impaired. Of the reel-to-reel tapes, 24 are identified by a letter.

The sequence is roughly alphabetical, though there are three distinct ones identified by the letter I, and there is none with the letter G, which could have been lost. 85 are identified with year spanning from 1973 to 1988. There are may duplicities, however, so the actual number of distinct recordings is likely much lower in this category. 10 have a numerical identifier, 29 have a textual identifier, and 2 have no identifier whatsoever. There is also a three-hour video on youtube [1].

3 Digitization

For cassettes, the digitization has mostly been done one side of a cassette to one file. For reel-to-reel, it was one channel of one pass from reel to reel to one file. Notable exceptions are cassettes digitized in auto-reverse mode, which has been experimented with during the two years of the digitization. An exhaustive list of the volumes corresponding to each digitized file follows:

1. sides of cassettes: 615 files,
2. whole cassettes: 140 files,
3. reel-to-reel passes: 112 files,
4. imported, uncertain: 222 files,
5. two concatenated cassettes: 1 file.

The imported files are those that have been digitized by other parties prior to my own effort. The format for digitization was 48 [kHz], 16 [bit], real time. An exception to real-time digitization were the three reels recorded in 2.38 [cm/s]. These have been digitized in the standard speed of 9.53 [cm/s] and had the sample rate set to one quarter of the nominal value.

[1] https://www.youtube.com/watch?v=UaNm9jnnJiA

4 Usage of Digitized Material

The digitized audio files have been used for two purposes. On one hand for direct distribution over physical media and on the other hand in a dedicated web application that has been serving for direct playback in the browser and for acquisition of manual corrections to ASR-generated [3] transcription. There were two generations of said web application, see Krůza 2012 [4] for the first one, and Krůza 2018 [5] for the second one. The original version used the HTML audio element for playback.

This suffers from poor timing precision when playing back specific words but it handles streaming very well. The next generation of the web application uses the Web Audio API, which remedies the precision issues at the cost of streaming capability. Initially, the second generation web application would load the whole recording, decode it in-memory and only then enable playback and other operation.

Since 90 minutes is a very common length of a recording (264 of the total 1090 files are over 80 minutes long), the load on the user computer was very heavy. A raw, decoded 90-minute monaural recording in 24KHz occupies about 240MB. However, after decoding a recording like that, the browser would often occupy triple that amount of memory. Even when the user's computer could handle that, the waiting time was in order of minutes, which ensured a very poor user experience. To remedy this issue, a change was necessary.

5 New Structuring

To enable the web application to access random segments of a long recording, while keeping the precision provided by Web Audio API, these options emerge:

1. pre-load and decode the whole recording,
2. serve ranges by cutting the audio on the server side,
3. store the recordings in smaller chunks,
4. wait for streaming to be supported.

Option 1 was our baseline as discussed above. Option 2 - serving arbitrary ranges cut on demand by the server seems quite viable. It is flexible and not very hard to implement. One downside is that the server must be able to script as well as access the whole recordings. Since the amount is in order of tens of gigabytes even using compressed audio formats, itlimits the options for hosting and increases costs. Another downside is with caching. Repeated sessions with the same recording would likely lead to different ranges being requested, thus disabling the advantages of caching.

Lastly, computation-heavy operations on the server side can impair site reliability or require costly cloud solutions. Option 3 - storing the corpus in smaller chunks has the downside of serving unnecessary context when requesting a specific range. Also, if we want to retain full download capability, we must either stitch the recording together or host both split and integral versions of all recordings. Option 4 - waiting for streaming support in Web Audio API might seem silly but the issue has been discussed by the Web Audio API developer team[2].

[2] https://github.com/WebAudio/web-audio-api/issues/1305

I opted for storing the corpus in smaller chunks. The downside of unnecessary context is a minor one, especially if we choose a fitting length of the chunks. Having to host two versions of the corpus raises the costs in my case by less than one US Dollar per month.

6 The Process of Restructuring

There are four choices to be made for the restructuring:

1. how long will the chunks be,
2. how to choose individual split points,
3. what will be the directory structure,
4. the file names.

6.1 Segment Length

The length of the chunks should be no more than 5 minutes. With 24kHz compressed audio, a 5-minute chunk takes up about 1.5MB. Theoretically, with modern 3G connections, even larger downloads should be instantaneous. However, the reality diverts from table speeds and we also have to wait for decoding, which takes about 4 seconds for a 5-minute chunk with an Intel Core2 @ 2.5GHz. According to the magazine UXMovement [8], four seconds are the threshold above which the user starts to abandon the original intent.

The Nielsen Norman Group [6] claims the same for ten seconds. The lower boundary is much freer but there always is a chance of an artifact at the glue point during playback, so the less of them the better. Also, each chunk means an extra HTTP request, which itself has considerable overhead. A reasonable compromise seems to be a span of 30 to 120 seconds for a chunk.

6.2 Finding Split Points

By choosing split points well, the impact of occasional artifacts caused by imprecisely switching segments during playback can be reduced. Ideally, splitting should be performed in pauses between sentences or at least between words. It is practically impossible to speak for two minutes without taking a breath, so there should always be a pause to be found for a segment's boundaries. There are numerous ways to find a moment of silence in an audio recording. The most requiring and most reliable way is manual annotation.

Another reliable method is by looking for predicted silences in phone-level-aligned transcription. This method could largely be used because there are manual or automatically-acquired transcriptions to most of the recordings in the corpus. Where either speech recognition or its forced alignment failed, and thus there is no transcription, another method comes to question: Voice-activity detection (VAD). I have used the perl module Audio::FindChunks for this purpose.

Table 1. Number of split points by their method of acquisition

acquisition method	number of uses
manually	0
by aligned transcription	60424
by voice activity detection	0
fix size	22043
total	82467

This method, unfortunately, is not very reliable in case of audio input with low signal-to-noise ratio, which is an abounding phenomenon in the Spoken Corpus of Karel Makoň. Where not even VAD helps, which can be identified so that the detected chunks are too long, there only remains the blunt method of creating fixed-size segments, disregarding the frequent case of splitting inside of a word.

Two of these methods have been eliminated by experiments: Manual annotation was way too inefficient. Six volunteers attempted this task, losing their patience after zero to ten minutes of transcribed material. Furthermore, detection by aligned transcription could be used for some recordings where initially speech recognition failed. By splitting the recordings into smaller chunks of fixed size and then recognizing each segment apart, the problematic recordings could be speech-recognized.

The resulting accuracy was catastrophic given the bad acoustic quality of these recordings but the longer silences were mostly precisely detected[3]. The chunks where the ASR failed again, were also beyond what VAD could handle, so there was no reason to resort to it. Table 1 summarizes the number of split points acquired by each method. The significant number of split points acquired by fix-size spliting is due to the fact that there are still recordings for which the split-and-recognize procedure is yet to be performed.

6.3 Split Point Selection

For fix-size segment splitting, I chose a segment length of 60 seconds. For splitting by aligned transcription, the task at hand is one where the input is a sequence of silences in the recordings, each identified by start and end, and the output is an optimal subsequence of these silences. The weight used is length of the silence (the longer the pause, the better it is as a split point). A constraint is the distance of the selected silences to be between 30 and 120 seconds.

Notice the task has no solution if the recording is shorter than 30 seconds. However, this never occurs in the Spoken Corpus of Karel Makoň and even if it did, we could simply leave the short recording in one segment. Another case where there is no solution is when the distance between adjacent silences (strictly speaking between the mid-points of adjacent silences) is greater than 120 seconds. Such a case occurs when the silences themselves are very long.

[3] There are no gold standard data, so there's no way of evaluating the result exactly but so far, I found no split in the middle of a word using this technique.

I have solved such cases manually but an automated approach would also be very simple. The algorithm sought seems to be a typical demonstration of dynamic programming: Find an optimal split of a part of the recording that only consists of the first word, then add next word and find the optimal split based on the current one. There is also a simpler variant, however, this algorithm is very easy to program and it has linear complexity.

1. start with the set of all silences;
2. iterate over them from shortest to longest;
3. remove the silence from the set if joining its adjacent segments doesn't yield one that's longer than 60 seconds;
4. end iteration;
5. iterate again over chosen silences;
6. remove a silence if one of its adjacent segments is shorter than 30 s;
7. end iteration.

6.4 File Naming

Naming of the chunks is arbitrary as long as the file names are unique. But a good naming scheme can help with further processing of the data. A good naming scheme can be kept when the split points are changed for a recording, without collisions. This excludes simple ordinal numbering. Another criterion is ease of filtering and parsing of the file names. Lastly, the if the file name is descriptive and human-readable, it can save time and effort. To accommodate these criteria. I chose to name the chunks as follows: `recording-identifier--from-start-time--to-end-time.format`

For example `87-25B--from-2186.45--to-2239.63.ogg`. To ensure smooth transitions, the segments are encoded 500 milliseconds longer than declared. This is to compensate for rounding of the length of the split file when encoded in a compressed format. Of course, this commands for the algorithm used in the web application to switch playback from one segment to another before the actual last sample of the expiring segment.

7 Reproduction

The corpus can be accessed through its web interface at[4]. The playback and other operations like playing back or downloading an arbitrary span (which can span a segment boundary) can be tested there. The source code for splitting the audio can be accessed at GitHub [5]. Similarly, the source code for the JavaScript part that works with the segments can be found at its GitHub repository [6], specifically in the files `src/store/audio.js` and `src/store/AudioChunks.js`.

[4] http://radio.makon.cz/

[5] https://github.com/Sixtease/CorpusMakoni/tree/master/scripts

[6] https://github.com/Sixtease/MakonReact

8 Conclusion

The new segmented structuring is used in the web application to emulate streaming capability while exploiting the advantages offered by Web Audio API. Other uses of the corpus, like direct downloads, archiving and on-demand forced alignment still rely on the original structuring where one recording of length in order of tens of minutes corresponds to a single file.

The redundant storage needs impose a negligible financial overhead but the work necessary for creation and maintenance of these extra files bring about an even stronger wish for the Web Audio API to support streaming. However, storing the corpus in files of a smaller size and devising the mechanism to seamlessly glue the chunks together during playback and transcription was a worthy lesson with potential use in other settings.

Acknowledgments. This work has been using language resources developed, stored and distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2010013). Participation in the conference was supported by COST action IS1404. This research was partially supported by SVV project number 260 104.

References

1. Crowdy, S.: Spoken corpus design. Literary and Linguistic Computing, vol. 8, no. 4, pp. 259–265 (1993)
2. Hájek, J.: Český mystik Karel Makoň. Dingir, vol. 2007/4, pp. 142–143 (2007)
3. Ircing, P., Krbec, P., Hajic, J., Psutka, J., Khudanpur, S., Jelinek, F., Byrne, W.: On large vocabulary continuous speech recognition of highly inflectional language-Czech. Seventh European Conference on Speech Communication and Technology, (2001)
4. Krůza, O., Peterek, N.: Making community and ASR join forces in web environment. International Conference on Text, Speech and Dialogue. Springer, pp. 415–421 (2012)
5. Krůza, O., Kuboň, V.: Second-generation web interface to correcting ASR output. Proceedings of the Future Technologies Conference (FTC). Science and Information Organization. Springer-Verlag, vol. 1, no. 1, pp. 749–762 (2018)
6. Nielsen, J.: Website response times (2010)
7. Reppen, R.: Building a corpus: what are the key considerations? The Routledge handbook of corpus linguistics. Routledge, pp. 59–65 (2010)
8. Tseng, A.: Progress bars vs. spinners: When to use which (2016)

ISSN 1870-4069

Development and Classification of a Chinese Humor Corpus

Yi-Ciao Gu[1], Yuen-Hsien Tseng[1], Wei-Lun Hsu[1],
Wun-Syuan Wu[1], Hsueh-Chih Chen[1,2]

[1] National Taiwan Normal University,
Taiwan

[2] National Cheng Kung University,
Taiwan

{minnie70011, chloe8599, greenteax23}@gmail.com
{samtseng, chcjyh}@ntnu.edu.tw

Abstract. In this work, we present a dataset for computational humor, which has the potential to be useful for Chinese e-commerce to enhance customer-machine/customer-clerk dialogue experience. The current humor corpus consists of 3,691 local jokes from more than 40 sources in Taiwan. Information retrieval technique is applied to remove near-duplicate jokes. The corpus is classified manually into 9 categories for potential use in proper context. Preliminary automated classification to discriminate the joke category using four traditional machine learning methods and three deep neural networks were conducted. The current results show that the performance of machine classifiers is far behind that of humans, leaving much room for research to apply them in proper context to achieve the goal of enhancing positive customer experience. Nevertheless, this developing humor corpus in traditional Chinese has indispensable value, because humor has at least subjective, cultural, regional, temporal, and linguistic characteristics, such that any local corpus has its value in the corresponding applications. Implication and potential application of this corpus are also discussed.

Keywords: Corpus annotation, joke classification, computational humor.

1 Introduction

Since Apple launched the Siri personal voice assistant in 2011, conversational user interface (CUI) has been gradually accepted by customers as the third human-machine interaction channel, in addition to the Web browsers and mobile Apps. Many companies have started to develop their own dialogue systems or chatbot platforms, allowing more businesses to provide a variety of active or passive customer services,

such as: booking, notification, promotion, etc. However, most CUI services are lack of humanity, which may lead to unsatisfactory experience during the interaction.

To improve customer's CUI experience, humorous dialogue can be applied as it has the potential to reduce customer complaints [1], to provide alternative effective responses outside the predefined services [2], and to win the trust from the customers. In addition, advertising, entertainment, and other industries are also the sectors to which humor is often applied for better products and services [3].

However, humor has at least five characteristics of being subjective, cultural, regional, temporal, and linguistic. To integrate humor into the conversational services, a local humor corpus is required to support the corresponding market. This work is aimed to develop a Chinese humor corpus for the mentioned purposes.

2 Related Work

In the field of natural language processing, human-computer interaction, and artificial intelligence, studies on humor identification and humor generation have been conducted since 1995, among which humor identification was considered to be more difficult than humor generation [4]. The goal of these studies is to explore humorous computational models, to enhance human-computer communication and user experience, or to assist people with communication disabilities to enhance their interpersonal interactions [5]. In recent years, advances in related technologies such as information retrieval, semantic processing, and machine learning, have further promoted the humorous dialogue research.

For example, the International Workshop on Semantic Evaluation (SemEval) held the Learning a Sense of Humor evaluation task in 2017 [6]. However, most studies focus on the humor identification research of English texts. As for the humor generation research, the use of English pun, acronym, and joke collection are the main approaches to generate humor. For humor classification study, Mihalcea and Strapparava [7] have collected 16,000 humorous and non-humorous one-liners, respectively.

They have tried stylistic features, such as alliteration, antonymy, and adult slang, to be used by decision trees, and found that alliteration is a better feature. They also use content words as features for Naive Bayes (NB) and Support Vector Machine (SVM), where SVM achieved 0.7751 accuracy in humor classification. Zhang and Liu [8] have manually collected 1,000 positive and negative humorous tweets. They use 50 features, grouped into 5 categories, for humor classification by Gradient Boosted Regression Trees, where special sign in tweets, ratio of content words, and polarity of terms are better predictors. Chen and Lee [9] use convolutional neural network (CNN) in comparison with the above traditional machine learning methods.

Their results show that CNN not only outperforms traditional methods, but also reduces the burden to design the features. In the SemEval 2017 task, the lessons learned are that humorous degree ranking is still difficult and that deep neural networks (DNN) normally perform better for the humor comparison task [6]. In humor generation study, humor and related corpora are indispensable resources. For example, Ozbal and Strapparava (2012) [10] combines English WordNet and ConceptNet resources with

Table 1. The metadata of the joke collection.

Field Name	Description
PKey	The primary key for identifying a joke
SourceTitle	Joke title from the source if any
GivenTitle	The first 10 characters of the joke content if SourceTitle is empty
Sharer	The one who share the joke, if known
Creator	The one who create the joke, if known
TextContent	The joke texts where line breaks are retained
PublicationDate	The publication date of the joke, if available
CollectingDate	The date when the joke is collected
SourceTopic	The topic/subject of the joke given from the source
GivenTopic	The topic/subject annotated by this article's authors
Length	The length of the joke content for easy of retrieving proper jokes
Language	Mostly in Chinese, a few are mixed with English and Taiwanese
SourceType	Name of the source (website, book, App)
Identifier	website URL, book ISBN, joke App name, etc.
Popularity	Humorous rating, number of likes, etc, if available

homophonic puns and metaphors to create creative names, especially humorous neologism.

Stock and Strapparava (2003) [11] used the incongruity theory to produce an interesting interpretation of the existing English acronym, e.g., reinterpreting technical words in religious terms, or providing users with concepts to generate interesting new words by the system. One of the examples is to convert IJCAI (International Joint Conference on Artificial Intelligence) into: Irrational Joint Conference on Antenuptial Intemperance.

3 Chinese Humor Corpus Development

There are a series of steps in developing this traditional Chinese humor corpus with sustainable and extendable value. These steps include: 1) select sources for collecting humorous jokes, 2) analyze the joke contents and define the fields (metadata) necessary for the corpus, 3) collect the jokes by various means, 4) remove near-duplicate jokes, 5) classify the jokes into categories for further use.

3.1 Joke Collecting and Cataloguing

To diversify the joke contents, we search and evaluate at lot of joke sources and, during a period of eight weeks, collect 3,828 jokes from 41 sources, which include 27 public websites (2777 jokes), 11 joke collection books (895 jokes), and 3 free Apps (156

jokes). After analyzing a sample of them, we decide to catalogue the joke collection based on Dublin Core Metadata Element Set [12]. The metadata finally contains 14 fields as listed in Table 1.

Some of the important fields include source identifier, source publication date, joke collecting date, sharer, author, and joke content. Note that it is hard to know the real author/creator or the copyright of a joke, as jokes may be told or revised in various ways before they are disseminated in social media or websites, and/or collected in books or Apps. These important fields are an attempt to provide as sufficient information as possible to trace back to the origin of the joke.

3.2 Duplicate Removal

As the jokes are accumulating, it is possible to collect duplicates from different sources. We then developed a full text matching tool, based on bag of words and TFxIDF term weighting, to detect near duplicates for removal [13]. This step reduced the number of jokes from 3,828 to 3,691, with each removal verified by the authors.

3.3 Manual Classification

A joke is humorous only when it is applied in a proper context. To help decide the right context, the corpus is manually classified into nine categories by at least two persons (all majored in Library and Information Science). Some of the 41 sources have their joke classification, especially for the website jokes. We adopt the most used categories and redefine their scope to yield the nine categories (the details are in the next section).

Two annotators classified each joke independently based on the category definition. When there is inconsistency (only 62 jokes with inconsistent categories), a third annotator helped label these jokes. Most of the categories among the three is then assigned to the joke. The inter-rater agreement for the two major annotators is as high as 0.97 in Cohen's kappa coefficient.

3.4 Corpus Format

The joke corpus is manually collected in an Excel file, with all field names in Chinese. We have written a Python code to convert it into an XML file with field names mapped to English as shown in Table 1. For manual update/development of the corpus, the Excel file is used as it is more efficient for manual editing. For computer processing, the XML file is recommended.

4 Discriminating Humor Categories by Machines

The high inter-rater agreement signifies the ease of discriminating the humor category by humans. The ability to automatically identify humors and their categories is of great value to later applications. As a preliminary exploration, we applied text classification techniques to evaluate the performance of machine humor discrimination. The

Table 2. Basic statistics of the joke corpus.

Joke	**Training Set**				**Testing Set**			
Category	Num	Avg	Max	Min	Num	Avg	Max	Min
C1: Lame	595	103	828	3	255	95	578	4
C2: Vocational	419	142	648	29	180	141	599	24
C3: Love-Affair	339	98	841	15	146	118	652	18
C4: Family	260	123	507	31	111	120	764	33
C5: Campus	245	123	759	14	105	116	293	18
C6: Adult	176	173	855	4	76	145	465	16
C7: Terminology	165	132	1140	14	70	130	1289	17
C8: Celebrity	155	107	518	12	67	129	713	18
C9: Others	229	107	1720	7	98	109	610	13

automated text classification follows these pipeline steps: 1) split of training/testing sets, 2) feature extraction, 3) model training, and 4) performance evaluation.

4.1 Training and Testing Sets

For training and testing machine classifiers, the corpus is split in a way that for each category 70% of the jokes are for training and the remaining 30% are for testing. Table 2 shows the nine categories with their number of jokes, and the average characters, maximum number of characters, and minimum number of characters for the jokes in a category.

4.2 Feature Extraction

To extract features for traditional machine classification, we currently exploit only the content words with four kinds of features. The text in each joke is first cleaned and standardized by tokenization, segmentation, and punctuation/stop word removal. Each text is then transformed into a feature vector with each element representing a term in the corpus. The term's value can be the term's occurring frequency (term frequency, TF) in a text, or the normalized TF multiplied by the logarithm of the inverse document frequency (IDF) of the term in the whole dataset (TFxIDF).

The term here can be a normal word, n consecutive words, or n consecutive characters. With these different values for a term and different ways to denote a term, there are four feature vectors (or feature sets) used: 1) Word Count: the term represents a normal word, and its value is the word's TF. 2) TFxIDF: the term is a normal word, and its value is the word's TFxIDF. 3) Word N-grams: the term is a N consecutive word in a text and its value is the term's TFxIDF. 4) Char N-gram: the term is a N consecutive character in a text and its value is the term's TFxIDF.

In contrast to the above sparse vectors where semantically similar terms may have no intersection in their vector representation, word embedding is a special way to

transform a word into a dense vector, where semantically similar terms are also highly similar in their embedding vectors [14]. Word embeddings can be trained using the input corpus itself or can be obtained from pre-trained word embeddings such as those pro-vided by fastText [15].

In our experiments, we used those provided by fastText, because fastText considers sub-word units such that out-of-vocabulary terms still have their embedding vectors by combining the corresponding ones of their sub-word units.

4.3 Design and Validation of Machine Classifiers

Four traditional ML methods are used, which are Naive Bayes (NB), Support Vector Machine (SVM), Random Forest (RF), and single hidden-layer neural network (NN) with a softmax layer as its output layer. For the deep learning (DL) methods, the first one is based on CNN, where pretrained word embedding with trainable weights for the application task is the first hidden layer with a dropout rate of 0.25 (same for all the dropout mentioned later). This is followed by a 1-d convolution layer to convolve the embedding vectors to extract local contextual information of the input word embeddings.

A global max pooling layer follows to summarize the local contextual information. A dense hidden layer with dropout is used to summarize the convolved/contextual messages into dense information. The final layer is then used to map the summarized information to a category prediction layer, which uses softmax as its activation function and is thus called a softmax output layer.

For the second DL method (RCNN), the first hidden layer has the same embedding structure as the CNN. The next layer is a bi-directional GRU (Gated Recurrent Unit) to extract longer dependent information in the text, which is followed by a 1-d convolution layer, a max pooling layer, and then a dense hidden layer with dropout.

The final output layer is a softmax layer. The third one is fastText [15]. Although it may not fully belong to a DL method, it combines some of the most successful concepts in natural language processing, such as word embedding and machine learning. It uses a hierarchical classifier instead of a flat structure, in which the different categories are organized in a tree. FastText also exploits the fact that classes are imbalanced by using the Huffman algorithm to build the tree used to represent categories.

The depth in the tree of very frequent categories is therefore smaller than for infrequent ones, leading to further computational efficiency [15]. All the above traditional classifiers adopt the default values of the Scikit-Learn package in Python. The training epoch for DL classifiers implement by Keras package with TensorFlow backend is set to 20 (the classifier sees each of the training texts for 20 times). With these settings, the DL classifiers achieve over 95% accuracy for classifying the training set in 10 epochs.

We think this training epoch is large enough to prevent the classifier from overfitting while restraining its training time. For fastText, we follow the tutorial instructions and try on various settings until we could not obtain better result. All the above classifiers

Table 3. Performance of various models with different features.

Models	Features	MicroF1	MacroF1	Time (seconds)
NB	Word Count	0.5027	0.4068	0.77
SVM	Char bi-gram	**0.5225**	**0.4503**	0.84
RF	Char bi-gram	0.4548	0.3601	1.09
NN	Char bi-gram	0.4990	0.4429	33.96
CNN	Word Embedding	0.4954	0.4102	922.02
RCNN	Word Embedding	0.4720	0.4144	4787.52
fastText	Word bi-gram	0.5036	0.4195	5.05

have been tested and verified on a balanced binary sentiment classification corpus[1], with the DNN approaches having slightly better performance (0.9967 for CNN vs 0.9948 for SVM in MicroF1 with 30%=2,126 test tweets).

4.4 Performance Evaluation and Results

Two metrics are used for performance comparison: MicroF1 and MacroF1. For MicroF1, a single confusion matrix for all the testing documents is calculated based on the ground-truth labels and the predicted labels. From this cross-tab matrix, precision and recall rates are calculated. Their harmonic average with equal weights to evenly emphasize both precision and recall is the so-called MicroF1 measure. For MacroF1, a confusion matrix is calculated for each category.

The F1 measures from each category are then averaged into MacroF1 with equal weights to emphasize each category evenly. Based on the above calculation, MicroF1 measures overall document classification effectiveness and thus reveals more the effectiveness of a few major categories. In contrast, MacroF1 takes each category's effectiveness into consideration and thus reveals more the effectiveness of most minor categories. In Table 3, we show only the best feature set for each classifier.

The results show that SVM is most effective with efficiency (all are run under a MacBook computer) far better than the DNN methods. The DNN methods do not show advantage like those discussed in the related work, even though they utilize pre-trained embedding vectors that exploit additional language knowledge. The large difference between 0.5153 MicroF1 and 0.97 inter-rater agreement implies that there is a large room for improvement, either in the feature extraction or in the learning model.

4.5 Error Analysis and Implication

Previous studies (as discussed in the Related Work) on humor classification use humorous jokes as positive examples and non-humorous similar texts as negative examples. The classification task is thus to discriminate whether a given text is humorous or not. While in our experiment, all testing texts are humorous jokes. Our goal in this article is to explore whether a machine classifier could tell which topic the

[1] https://www.kaggle.com/c/si650winter11/data

Table 4. Confusion matrix for the SVM classifier.

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	
C1	**159**	25	18	8	6	5	10	6	18	255
C2	33	**121**	7	5	0	3	1	3	7	180
C3	32	7	**89**	2	2	8	2	0	4	146
C4	16	3	5	**78**	1	3	0	2	3	111
C5	8	5	2	1	**81**	2	0	2	4	105
C6	24	11	11	3	6	**11**	1	3	6	76
C7	27	6	10	3	8	0	**10**	1	5	70
C8	23	9	5	4	4	0	4	**14**	4	67
C9	31	15	18	3	4	2	4	5	**16**	98
	322	187	147	104	108	32	28	31	51	1010

Table 5. Breakdown performance for the SVM classifier.

	Precision	**Recall**	**F1-score**	**Support**
C1: Lame	0.45	0.62	0.52	255
C2: Vocational	0.60	0.67	0.63	180
C3: Love-Affair	0.54	0.61	0.57	146
C4: Family	0.73	0.70	0.72	111
C5: Campus	0.72	0.77	0.75	105
C6: Adult	0.32	0.14	0.2	76
C7: Terminology	0.31	0.14	0.2	70
C8: Celebrity	0.39	0.21	0.27	67
C9: Others	0.24	0.16	0.19	98

input joke (e.g., uttered by a user) is about, such that the subsequent response by the machine remains in the right context.

Table 4 lists the confusion matrix for the best classifier, SVM. It can be seen that most texts in small categories have been incorrectly classified into larger categories. For example, 23 texts in the C8: Celebrity are classified into C1: Lame jokes. This tendency has already been indicated by the lower MacroF value (lower than MicroF). However, there are two medium size categories that exhibit relatively high performance: C4: Family and C5: Campus jokes, as shown in Table 5, implying that the term variations are smaller between the training set and the testing set. This indicates the feasibility of applying machine classifiers to these topics. Recently, there are better deep learning models for NLP, such as BERT from Google [16] and GT2

from OpenAI [17]. They may be of great help to improve the classification performance. However, the difficulty to be able to interpret the reason (compared to the salient features revealed by SVM) may hinder the understanding of the mechanism of computational humor.

5 Potential Applications of the Corpus

In a conversational user interface (CUI), understanding the user's context, intent, or topic is valuable to create an empathetic CUI system. Despite higher prediction performance is observed in other corpora, the above experiments show that there is much to explore to achieve this goal, where topic-dependent solutions may be useful to improve the overall performance. Our corpus could contribute to this line of study. In contrast to the above humor discrimination or identification task, another line of computational humor is about humor generation.

An example of applying this corpus for this purpose is providing a CUI for users to retrieve relevant jokes. We have built a chatbot in the LINE platform (a popular mobile messaging App in Eastern Asia), named as IceBreaker, to allow a nervous-prone user to retrieve a joke for telling in front of a group of people when he/she needs to. The user may say to the IceBreaker (using the built-in voice recognition in a cell phone), like: "give me a joke about joke" (or "tell me a joke about *some_topic/some_keyword*", or simply input a keyword).

The IceBreaker would respond with some texts like: "Why can't you tell a joke at the beach? Because there would be a Tsunami!" Here Tsunami in Chinese sounds like "the Sea is laughing". The joke reminds the user to apply some homophone skills to tell a joke that may lead to some devastating consequence (which complies with the incongruity theory in humor study). The chatbot also records the feedback from the user to accumulate the average degree of the humor of the joke. The user can also feedback a new joke through the chatbot. As more information like these is accumulated, the corpus, as a seed, would be expanded and refined by its users.

6 Conclusions and Future Work

We have presented a manually collected and annotated local humor corpus, which is the first traditional Chinese one to our best knowledge. The corpus together with the tools for near-duplicate detection, category classification, and joke retrieval via LINE chatbot will be made public once we figure out the copyright issue.

We wish that it would be useful and extendable for future research and also applicable in practical application environment. For linguists and those social researchers who study humor as an important issue, we expect this manually catalogued corpus to be a valuable and sustainable dataset for further exploration.

Acknowledgments. This work is supported in part by the Ministry of Science and Technology under the grant MOST 107-2221-E-003-014-MY2 and MOST 108-2634-F-002-022, and by the "Institute for Research Excellence in Learning Sciences" of

National Taiwan Normal University from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education in Taiwan.

References

1. Binsted, K.: Computational humor. IEEE Intelligent Systems, vo. 21, no. 2, pp. 59 (2006)
2. Bellegarda, J. R.: Spoken language understanding for natural interaction: The siri experience. natural interaction with robots, knowbots and smartphones. New York, USA, Springer (2014)
3. Mihalcea, R., Strapparava, C.: Technologies that make you smile: adding humor to text-based applications. IEEE Intelligent Systems, vol. 2, no. 5, pp. 33–39 (2006).
4. Zhang, R., Liu, N.: Recognizing Humor on Twitter. In: The 23rd ACM International Conference on Information and Knowledge Management. Shanghai, China, ACM, pp. 889–898 (2014)
5. Ritchie, G., et al.: The STANDUP Interactive Riddle-Builder. IEEE Intelligent Systems, vol. 21, no. 2, pp. 67–69 (2006).
6. Potash, P., Romanov, A., Rumshisky, A.: SemEval-2017 Task 6: #HashtagWars: learning a sense of humor. In: 11th International Workshop on Semantic Evaluations, Vancouver, Canada (2017)
7. Mihalcea, R., Strapparava, C.: Learning to laugh (automatically): Computational models for humor recognition. Computational Intelligence, vol. 22, no. 2, pp. 126–142 (2006)
8. Zhang, R., Liu, N.: Recognizing humor on Twitter. In: 23rd ACM International Conference on Information and Knowledge Management, Shanghai, China (2014)
9. Chen, L., Lee, C. M.: Predicting audience's laughter using convolutional neural network. arXiv:1702.02584 (2017).
10. Ozbal, G., Strapparava, C.: Computational humour for creative naming. In: 3rd International Workshop on Computational Humor, Amsterdam, Netherlands (2012)
11. Stock, O., Strapparava, C.: Getting serious about the development of computational humor. In: 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico (2003)
12. Dublin Core Metadata Element Set, Version 1.1: Reference Description (2012) Retrieved from http://dublincore.org/documents/dces/.
13. Tseng, Y. H., Teahan, W. J.: Verifying a Chinese Collection for Text Categorization. In: 27th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '04, Sheffield, U.K (2004)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems (2013)
15. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. CoRR, abs/1607.01759 (2016)
16. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2018)
17. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language Models are Unsupervised Multitask Learners (2019)

ISSN 1870-4069

Is There a Linear Subspace in Which the Difference Vectors of Word Analogy Pairs are Parallel?

Stephen Taylor[1], Tomáš Brychcin[1,2]

[1] University of Western Bohemia, Pilsen CZ,
Czech Republic

[2] SentiSquare,
Czech Republic

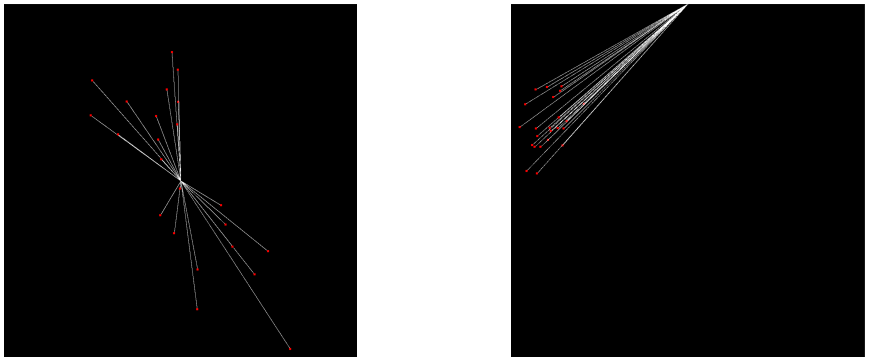
stepheneugenetaylor@gmail.com,
brychcin@sentisquare.com

Abstract. Since Mikolov introduced word analogies as an example of semantic composition by vector addition, they have inspired both enthusiasm and disdain. If the arithmetic computation works, the relationship encoded in the word vectors should manifest itself as parallel difference vectors, and if the difference vectors are parallel, this should appear in two-dimensional projections. For Principal Component Analysis (PCA) bases computed on just the words of a relation's pairs, this seems to be true. However, PCA on larger subsets of the vocabulary typically shows a wide range of directions for difference vectors in the same relation. The PCA phenomenon is evidence for our suggestion that there is a subspace for each relation, in which the difference vectors are parallel. That is, only a subset of the semantic information for each word participates in the relation. To approximate such a subspace, we train a linear transformation which moves a portion of the pairs in a relation so that the difference vectors are nearly parallel to each other, while minimizing the movement of unrelated words. We see that there is a net improvement in evaluating not only analogies which include pairs in the training set, but also analogies between held-out pairs in the same relation. The trained transformation thus seems to isolate semantic components expressed by the relation.

Keywords: Word analogies, word vector semantics space, semantic composition.

1 Introduction

Mikolov et al. [11] introduced solving word analogies with vector arithmetic in the same paper that introduced **word2vec**, the algorithms and software release for rapidly constructing CBOW and Skip-gram semantic vector spaces. Word analogies fired the imagination, because they are similar to some standard questions on human intelligence tests [17]. Briefly, Mikolov [11] asserted that if $\mathbf{s}(word)$ is the vector for $word$, then the word analogy *Man:King :: Woman:?* can be solved for *?* = *Queen* by finding the word closest in the vector space to $\mathbf{s}(\textit{King}) - \mathbf{s}(\textit{man}) + \mathbf{s}(\textit{woman})$. They provided a corpus of word analogies which test this idea, now called the Google word analogy corpus, which we use in this paper.



(a) PCA uses whole vocabulary **(b)** PCA uses only relation file

Fig. 1. Various 2-D projections of `capital-common-countries`, using PCA on different sets of vectors.

This computation scheme is equivalent to the claim that in the semantic vector space, vector addition is equivalent to semantic combination, a claim made later that same year by Mikolov [12].

In Figure 1, we show two different PCA projections of the difference vectors for the capital-common-countries relation into a two-dimensional vector space. The difference vectors are all aligned to start from the origin, which brings out the parallelism, or lack of it, sharply. Because each figure has a different set of basis vectors, each rosette is aligned and centered differently. The difference vectors are closest to parallel in Figure 1B, in which the eigenvectors, or axes of the projection, are computed based only on the 23 distinct pairs, or 46 words, in the relation.

In general, the difference vectors in analogy files (hereafter *relations*) are not parallel in 300 dimensions [9,15] but can look more aligned after projection to two dimensions, particularly if the PCA is done on a subset of the data. If their difference vectors were actually parallel and of the same length, then word analogies would have 100% evaluation success, which would make them formidable tools, instead of the interesting demonstrations which they now are.

Although none of the PCA projections captures much of the variation, they make it seem plausible that there might be a linear transformation of the vector space which minimizes information irrelevant to a particular relation. We set ourselves the goal of finding, for each relation, a linear transformation such that the difference vectors in the transformed space would be parallel.

2 Related Work

Coecke et al. [1] proposed building an algebra for composing word vectors into phrases. Mitchell and Lapata [14] consider how vectors in a semantic space, in their case Latent Semantic Analysis (LSA) vectors describing usage context for individual words, might be composed into phrases.

They asked human beings to rate phrase similarity for two-word phrases, and attempted to correlate the human similarity scores with scores computed using nine different *composition functions* applied to the LSA vectors of the individual words. They found the best correlation for element-wise multiplication. Vector addition, as proposed by Mikolov et al. [11,12] was in the middle of the group of tested composition functions. Levy et al. [7] make an early investigation of vector-space word analogies; their article includes two versions of the vector-addition [13] method, and a new scheme of their own.

They exhibit a very-high dimension word embedding, in which context counts are dimensions, to explain how vector addition could plausibly perform semantic composition. They describe all the algorithms as forms of vector similarity calculation. In [8] Levy examines various examples of hypernym-analogies, and concludes that what they measure is not whether the analogy is valid, but whether or not a word is a hypernym of *something*. Linzen [9] argues that frequently the difference vector is irrelevant in analogy evaluation, and that a nearest neighbor effect overwhelms it.

Drozd et al. [2] notes that averaging the difference vector over many pairs makes it more reliable, and that then using logistic regression to determine a class for second words in a pair for the relation and checking whether a proposed answer is in the class gives excellent results. They tested their algorithm using two held-out pairs in the relation. This work has more emphasis on 'solving' analogies than ours, but does consider smoothing difference vectors to be worthwhile.

Finley et al. [3] test whether the vector-arithmetic strategy (without exclusion) is an improvement on word similarity on several test-sets. They concluded that some relations don't work well, but several do; they categorize the most successful analogy types as *Inflectional Morphology*; *Named Entities*; *Gendered Nouns*. Least successful types are *Derivational Morphology* and *Lexical Semantics*.

Gittens et al. [4] consider semantic combination, as a mathematical operation on word vectors in a syntactic space generated with the Skip-gram algorithm.

They provide a model for paraphrases in terms of the target and context vectors used in Skip-Gram algorithm, which leads to a formula for finding them, and conclude that when Skip-gram is applied to a corpus with a uniform word distribution, semantic combination is vector addition.

Natural language words follow a Zipf-like distribution, but Finley [3] note that analogies in which all four words have similar frequency seem to work better.

Vylomova et al. [18] train linear kernel SVMs to determine from their difference vectors whether pairs were members of a relation or not.

They find using negative samples in the training set improves precision but lowers recall. Konkol et al. [6] trained a linear transform from place-names to geographical co-ordinates, demonstrating that vector semantic spaces apparently have some location information encoded in them.

Szymanski [16] obtained a similar result with a different method, by training word vectors with parallel corpora from specific time periods in order to look for equivalents such as Ronald Reagan in 1987 is like Bill Clinton in 1997.

Table 1. Some statistics on the Google word analogies.

relation	analogies	pairs	base correct	base spares	base accuracy
capital-common-countries	506	23	179	236	0.35
capital-world	1482	39	380	753	0.26
city-in-state	1560	40	304	942	0.19
country-currency	342	19	45	109	0.13
family	506	23	177	253	0.35
gram1-adjective-adverb	992	32	17	271	0.02
gram2-opposite	756	28	15	300	0.02
gram3-comparative	1332	37	501	687	0.38
gram4-superlative	1122	34	234	699	0.21
gram5-present-participle	1056	33	64	771	0.06
gram6-nationality-adjective	1482	39	1166	203	0.79
gram7-past-tense	1560	40	161	871	0.10
gram8-plural	1332	37	67	1153	0.05
gram9-plural-verbs	870	30	136	462	0.16

3 Methodology

3.1 Public Data

We downloaded a semantic vector space file[3] and edited it to keep only the first 150,000 words.The vectors for this file have 300 32-bit floating point elements. We edited the Google analogy dataset[4] to break the analogies into pairs, which lets us consider training- and test-sets from among the pairs in each analogy type, which we then call a *relation*. The fourteen relations of the Google analogy set, and a few statistics about each are shown in Table 1.

Some of the relations have fewer pairs than appear in the downloaded test set; this is because pairs with out-of-vocabulary words are eliminated. This occurs because we limited the size of the vocabulary to 150 thousand words. These are the most frequent words in the Google news corpus from which the word vector space was built.

The downloaded file has vectors for about 3 million word-forms, but searching through so many vectors to find a near-match takes 20 times as long as searching through the smaller list. Also, using a restricted vocabulary removes some competition for the target word, perhaps resulting in higher reported accuracy. The **analogies** column in Table 1 is computed from the number of pairs as:

$$\text{analogies} = \text{pairs}(\text{pairs} - 1). \tag{1}$$

Thus for any two different pairs, we can make two analogies. For some analogy types we could make four by reversing the pairs, but we do not do this. For example the analogy (from the `gram8-plural` relation):

$$\text{mouse} : \text{mice} :: \text{cat} : \text{cats}. \tag{2}$$

[3] GoogleNews-vectors-negative300.bin.gz from https://code.google.com/archive/p/word2vec/

[4] http://download.tensorflow.org/data/questions-words.txt

could be manipulated to put any of the four words in the final position that the algorithm calculates. However, this is not obviously true for the analogy (from the `city-in-state` relation):

$$\text{Fresno : California :: Tucson : Arizona.} \tag{3}$$

because if we provide the state, there seem to be many equally good answers for a city in the state. The **base correct** column is the number of analogies for which vector addition provides the expected answer as the first choice. Mikolov's version of the algorithm excludes the three other words in the analogy from being considered.

We provide the column **spares** to record when the correct answer would be provided by this work-around. Although word similarity is interesting, it has nothing to do with phrase composition by vector addition. A glance at the table shows that spares are a major contributor to the usual statistics for these relations.

3.2 Transforming the Semantic Space

The semantic vector space $\mathbf{S}$ is defined by a function, where W is a set of words:

$$\mathbf{s}(w) : W \to \mathbb{R}^n. \tag{4}$$

In the case of our downloaded `GoogleNews-vectors-negative300`, each record in the file contains one string and an associated 300-element vector, and the association between those strings and vectors defines $\mathbf{s}(w)$. The strings in the records define the domain W of $\mathbf{s}(w)$ and the vectors are the values. The order of records is not important to the definition of the function, but they are partially ordered by frequency of words in the corpus.

This is convenient for editing the function to omit rare words, as we do. Our function $\mathbf{s}()$ can thus be represented a mapping from words to an index, and a $150000 \times n$ array $i\mathbf{D}$. We can create a 'transformed' space $\mathbf{T}$ with m dimensions by building a function $\mathbf{t}(w) : W \to \mathbb{R}^m$, using the same word mapping as for $\mathbf{s}()$ and a new dictionary array, where C is an $n \times m$ array. We then call $\mathbf{T}$ a linear subspace of $\mathbf{S}$:

$$\mathbf{D}' = D \times C. \tag{5}$$

3.3 Evaluating analogies

An important function for numerical computation of word analogies is

$$\text{neighbors}_{\mathbf{S}}(v, n, m) : \mathbb{R}, \mathbb{N}, (\mathbb{R}^n, \mathbb{R}^n \to \mathbb{R}) \to W^n, \tag{6}$$

where v is a point in $\mathbb{R}^n$, n is a small integer, and m is a metric function. The $\text{neighbors}_{\mathbf{S}}()$ function obviously needs access to the dictionary function $\mathbf{s}$ in order to find a word from a point. The function $\text{neighbors}_{\mathbf{S}}(v, n, m)$ returns an ordered list L of n words $w_i \in W$ such that $\mathbf{s}(w_i)$ are closest to v according to m, that is

$$\forall(x \in W)\forall(w_i \in L)(m(x, v) < m(w_i, v)) \implies x \in L. \tag{7}$$

We denote pairs in a relation as p_i, and the two ordered elements of the pair as p_i^0 and p_i^1. Each pair has a difference vector d_i:

$$d_i = \left(\mathbf{s}(p_i^1) - \mathbf{s}(p_i^0)\right). \tag{8}$$

To compute the value of ? in in the analogy

$$p_i^0 : p_i^1 :: p_j^0 : X, \tag{9}$$

we evaluate

$$\text{neighbors}_{\mathbf{S}}(\left(d_i + s(p_j^0)\right), 1, \text{cosd}) = p_j^1, \tag{10}$$

where cosd is the cosine distance between two vectors. We hope to find that $X = p_j^1$.

3.4 Finding a Transform

Our hypothesis is that for each different relation there is a non-zero matrix $\mathbf{C}$, such that for any two pairs in the relation:

$$\left(\mathbf{s}(p_i^1) - \mathbf{s}(p_i^0)\right) \times \mathbf{C} \approx \left(\mathbf{s}(p_j^1) - \mathbf{s}(p_j^0)\right) \times \mathbf{C}, \tag{11}$$

that is

$$d_i \times \mathbf{C} \approx d_j \times \mathbf{C}. \tag{12}$$

In testing, we quantify $\approx$ in equation 11 and 12 to mean that the accuracy of solving the analogy (in the space $\mathbf{T}$ which is $\mathbf{S}$ transformed by $\mathbf{C}$) correctly is some fraction close to one. That is, we judge our success in finding C by the fraction of instances (i, j) in the relation which which satisfy Equation 10, but using the $\mathbf{T}$ space, instead of the $\mathbf{S}$ space.

We are working with 300-dimensional vectors, and we have from 19 to 40 pairs in each relation. We need to hold out at least one pair for testing; we experimented with holding out 2,4,6,8,10,14, and 16 pairs. The rest of the pairs in the relation are the training set. We want to train the values in C so that the difference vectors between the words in the pairs in the training set are transformed by the matrix multiplication into the mean difference vector for the training set.

Before training we place the difference vectors for the pairs in the training set in a matrix $\mathbf{D}$ and their (identical) targets in a target matrix T. We experimented with using both difference vectors and word-vectors with modified locations for training, but difference vectors give better results. The problem with this, is that a perfectly good solution could project all of S along the relation mean, densely packing the vector with points irrelevant to the relation.

Ideally, word points in the vector space for words not in the relation would either stay put, or shift without condensing. To encourage this to happen, we tried a strategy of *pinning* difference vectors. We add *pinned* vectors to the $\mathbf{D}$ and $\mathbf{T}$ matrices. These are a number of difference vectors between words chosen at random, which we want not to change; that is, we assume that a randomly chosen word pair (w_i, w_j) is not a pair of the relation, or is a *negative example*. For these vectors, the rows in $\mathbf{D}$ and $\mathbf{T}$ are the same, $\mathbf{s}(w_i) - \mathbf{s}(w_j)$. One of the nice benefits of training difference vectors

instead of training the individual words of a pair, is that we can be more confident that two randomly chosen words are not part of the relation we are training for. In the case of syntactic relations like verb-past-tense, a randomly chosen word is quite likely to be a verb, and potentially part of the relation, even though it might not be part of the provided examples. We train the matrix C with gradient descent, a learning rate of 0.001, for 3000 iterations, with a regularization constant of 0.98. Then, after training,

$$\mathbf{T} \approx \mathbf{D} \times \mathbf{C}. \tag{13}$$

A final part of the training goal is a regularization step. The objective function consists of two terms:

1. The sum of squares of each coordinate in $(\mathbf{T} - \mathbf{D} \times \mathbf{C})$.
2. A regularization term ρ. Considered as part of the objective function, this relates to a constant times the sum of the squared elements of the $\mathbf{C}$ array.

However, all we need during training is the regularization constant, $\rho \leq 1$, and the partial derivative δ_1 with respect to the $\mathbf{C}$ array, which we compute as:

$$\delta_1 = (2 * (\mathbf{D} \times \mathbf{C}) - \mathbf{T})^\top) \times \mathbf{D}. \tag{14}$$

and apply at each training step as:

$$\mathbf{C}' = (\rho)\mathbf{C} - (\text{LearningRate})\delta_1. \tag{15}$$

3.5 Evaluating the Analogies

In previous work with analogies, we have used two different kinds of normalization, in which every vector in the space $\mathbf{S}$ is changed. *Zero-centering* first computes the mean of all vectors in the space, then subtracts the mean from every vector. As a result, the new mean of each vector coordinate is zero. This translation does not affect the angles between difference vectors in the space. However angles with the new origin are different than angles with the old origin were, and points which were on a single line extending from the origin no longer are.

Thus the nearest neighbors of points in space as computed using cosine distance may change (Euclidean distances would not change). *Unit normalization* computes the square root of the sum of the squares of the coordinates of a vector, that is, the Euclidean distance to the origin, and divides all the coordinates by this number, effectively moving all points to the surface of a 300-dimensional hyper-sphere at unit distance from the origin. This transformation does not effect cosine distance, and it enables us to compute the cosine distance without recomputing vector norms, but it does change difference vectors.

The vectors between points on the surface of the sphere now point off into empty space when applied to other starting points. Of course, for sufficiently short vectors, they might not point *far* off the surface; but the angle from a word to its nearest neighbor is typically near $\pi/4$. This would put the calculated answer approximation for word analogies $\sqrt{(2)} - 1$ off the surface of the hyper-sphere, changing Euclidean distances (but not their relative ranking) to nearby words.

Furthermore the notion of parallel vectors on the surface of the sphere works only for small neighborhoods, so it is not obvious how to make use of difference vectors, let alone retrain them. In spite of these concerns, using these normalizations has apparently helped the success rate of analogy evaluation in the past. It has been suggested that just as unit normalization of vectors in the Information Retrieval vector space model [10] compensates for long documents, unit normalization of word vectors compensates for word frequency.

But since in this study we are specifically concerned with difference vectors, and difference vectors are impacted by normalization (and in previous studies difference vectors are not the main factor in accuracy evaluations, as may be seen in Table 1) we elected not to normalize for these experiments. Furthermore, Mikolov's word-search policy for finding the last word in the analogy explicitly rejects returning any of the first three words. This disallows some kinds of analogies, for example

$$\text{Prince Harry} : \text{Queen Elizabeth} :: \text{Prince William} : ? \tag{16}$$

where the answer is one of the guiding words, but more importantly, Linzen [9] points out that if the right answer is closer to the guide words than the computed vector,the computation and the difference vector are irrelevant, and only word similarity is being tested. So we don't eliminate the other words in the analogy from consideration, and we consider the answer wrong if the nearest neighbor of the computed vector turns out to be one of the other words in the analogy, even if the correct answer is closer than any other word except for the guide words.

We do keep track of this situation, primarily to able to compare the baseline figures with those of other researchers. We call this situation a *spare*, after the play in bowling in which some pins remain after the first ball, but are all knocked down by the second ball.

4 Experimental Results

The bar charts in Figure 2 and Figure 3 illustrate that the transformation improves performance on the word analogies for all of the four possible choices of pairs chosen from the training set andthe held-out, untrained pairs. This suggests that the transformation might actually accomplish the goal of isolating the semantic component of the relation that it is trained on, including on words on which it is not trained. Because the relations are small, and the chart is drawn from a particular training run, we see quite dramatic small number effects; the base accuracies between the four groups differ by large percentages, as do the trained accuracies.

The capital-common-countries relation is one of the best-performing of the Google set, but our statistics are lower than others, because we don't give credit for *spares*. We have a base performance before training of 179 successes for 506 analogies, or 0.35. We also note 236 spares, which would give a total accuracy of 0.82, but the spares are a measure of word-similarity, not of success in the difference vector calculation. Table 2 shows experiments working out the improvements on the held-out set for various parameter values, in this case for pinned words instead of difference vectors. Both Figure 2 and Figure 3 use 8 pairs held out and 50 pinned difference vectors.

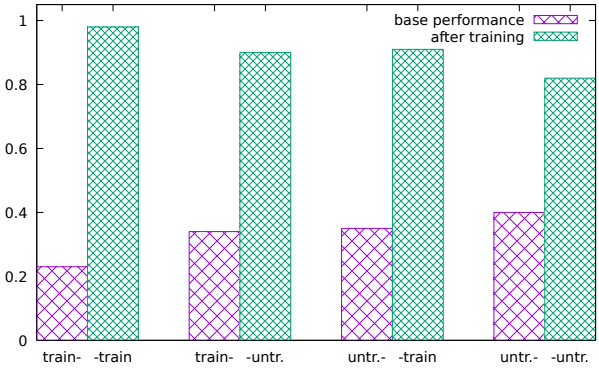


Fig. 2. Accuracy for capital-common-countries relation before and after trained transformation.

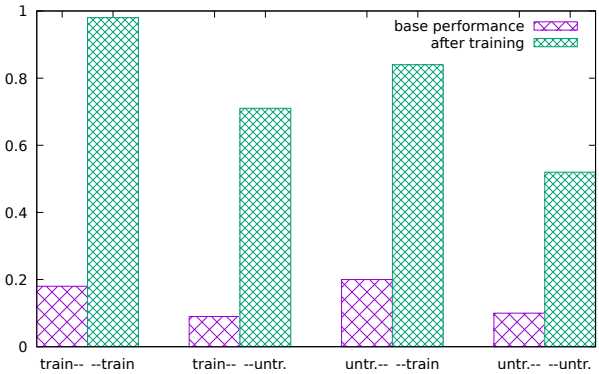


Fig. 3. Accuracy for `gram9-plural-verbs` relation before and after trained transformation.

The capital-common-countries relation has 23 pairs, and the held-out row labels at the left of the table show how many pairs were held out for testing. In the top row, four pairs are held out, so nineteen were used for training, and in the bottom row only seven pairs are available for training. The column headings show the number of pinned points, or negative examples, that are used in training. The best run in this table shows an absolute improvement of 0.16 in the fraction of correct results, while the surface plot shows the total accuracy. We built tables like Table 2 for all the relations, and then repeated the exercise focusing on the regions with the best results.

Table 3 shows the results of runs using the best set of parameters for each of the relations. In this table, the names of the relations are abbreviated to fit the table on a page, and several descriptive numbers are squeezed into the second column. The first two numbers are *pairs*, the number of pairs in the relation, and *analogies*, the number of analogies that can be built with the given number of pairs (always $pairs(pairs-1)$). The last two numbers in the second column are the training parameters: *held* is the number of pairs held out for testing, so

$$train = pairs - held. \tag{17}$$

Table 2. Improvements in analogy performance between untrained pairs in the capital-common-countries relation after training.

	0	**10**	**50**	**100**	**200**	**500**
4	0.00	0.00	0.08	0.08	0.08	0.00
6	0.07	0.10	0.13	0.13	0.13	0.03
8	0.02	0.07	0.14	0.16	0.09	0.05
10	0.01	0.09	0.10	0.13	0.10	0.04
12	0.00	0.05	0.12	0.10	0.07	0.04
14	-0.04	0.02	0.09	0.11	0.08	0.03
16	-0.05	0.03	0.11	0.13	0.09	0.05

Table 3. Some selected results for the google relations. tp=trained pair; hp = held-out pair.

Relation	pairs/analogies/ held/pinned	base accuracy	base accuracy with spares	trnd accuracy	tp: :hp base	tp: :hp trnd	tp: :hp base	tp: :hp trnd	hp: :tp base	hp: :tp trnd	hp: :tp base	hp: :tp trnd
capital-common...	23/506/14/150	35%	82%	89%	23%	98%	34%	90%	35%	91%	40%	82%
capital-world	39/1482/4/150	25%	76%	97%	25%	99%	30%	89%	23%	91%	25%	66%
city-in-state	40/1560/4/200	19%	79%	96%	20%	98%	10%	88%	18%	95%	0%	75%
country-curren...	19/342/6/100	13%	45%	64%	13%	92%	12%	44%	16%	48%	3%	13%
family	23/506/12/50	34%	84%	71%	56%	99%	31%	70%	34%	74%	20%	46%
gram1-adjectiv...	32/992/8/100	1%	29%	71%	0%	91%	2%	49%	2%	50%	5%	19%
gram2-opposite	28/756/6/100	1%	41%	84%	1%	95%	3%	53%	2%	83%	6%	50%
gram3-comparat...	37/1332/6/200	37%	89%	97%	42%	98%	31%	92%	23%	97%	13%	93%
gram4-superlat...	34/1122/10/150	20%	83%	92%	17%	99%	24%	88%	22%	92%	27%	58%
gram5-present-...	33/1056/10/50	6%	79%	83%	6%	99%	5%	68%	5%	78%	8%	48%
gram6-national...	39/1482/16/150	78%	92%	96%	80%	96%	73%	97%	80%	96%	79%	95%
gram7-past-ten...	40/1560/10/50	10%	66%	85%	10%	95%	9%	62%	11%	87%	11%	53%
gram8-plural	37/1332/4/50	5%	91%	97%	5%	99%	1%	84%	9%	97%	0%	75%
gram9-plural-v...	30/870/12/100	15%	68%	81%	18%	98%	9%	71%	20%	84%	10%	52%

Is the number of difference vectors from the relation in the training set; and *pinned* is the number of other difference vectors added to training set to stabilize it. Columns three and four of Table 3 also appear in Table 1. Column three, *base accuracy* is the fraction of analogies in the relation which are correctly solved with vector arithmetic, with our conditions: The vocabulary is restricted to only 150000 words, so that the number of words within any radius of the computed point is reduced, increasing slightly the likelihood that the nearest one is the given solution; and we insist that the word nearest the computed point is the only one considered (no spares).

Column four compares the accuracy when spares are allowed, thus partially answering the question "How could anyone think that a relation with a 1% accuracy rate is semantically interesting?" Column five is also a partial answer to that last question. It shows the accuracy after training and for every relation except `family`, it is higher than the traditional figure in column four, suggesting that with a little squeezing you can find some common semantics even if the concentration was not originally very high.

We can use *train* and *held* to compute the size of the four sub-relations described in the last eight columns of Table 3:

– **tp::tp** consists of word analogies constructed between pairs both of whose

difference vectors were in the training set. The size of this sub-relation is $train(train - 1)$.

- **tp::hp** consists of word analogies constructed with a pair from the training set on the left-hand side, and a pair from the held-out set on the right-hand side. The size of this sub-relation is $train(held)$.
- **hp::tp** consists of word analogies constructed with a pair from the held-out set on the left-hand side, and a pair from the training set on the right-hand side. The size of this sub-relation is also $train(held)$.
- **hp::hp** consists of word analogies constructed only with pairs from the held-out set. The size of this sub-relation is $held(held - 1)$.

For each of these sub-relations we show the base performance, that is, accuracy before training, and the trained performance. It is not a surprising result that after training, the performance of analogies from pairs in the training set (column 7, **tp::tp**, **trnd**) is very good. The lowest result is for the `gram1-adjective-adverb` relation, 91%.

However, it is interesting the sub-relations with mixed pairs from the trained and held-out sets (columns 9 and 11) also perform well; the lowest performance is for the `country-currency` relation, 44%, slightly lower than `gram1-adjective-adverb`, 49%. The two sub-relations seem to have similar performance, with many results clustered around 90%. The most interesting columns are the last two, in which we see that the accuracy of relations constructed with the held-out pairs has increased in every case.

The average accuracy rises from 18% in column 12 to 59% in column 12. These results seem to show that it is possible to build a linear transformation which isolates some of the semantic information whichpt a set of analogies depends upon. Two relations seem to generalize to untrained pairs particularly poorly, `country-currency` and `gram1-adjective-adverb`. Both also have very low base accuracy, and it seems possible that the relations just don't work; either the information is not in the semantic space, or it is stored in such a way that a linear transformation cannot enhance it.

Both seem to have little room for human error in building the relations – although perhaps currency is in flux, with the still on-going adoption of the Euro. An additional point raised, in particular by the contrast between base accuracy and the much larger accuracy with spares, is to what extent the Google relations include spurious pairs, not just from the point of view of additive semantics, but in terms of their similarities to each other. For example, the analogy

$$\text{man} : \text{woman} :: \text{King} : \text{Queen}. \tag{18}$$

Is solved by Mikolov's technique because 'King' and 'Queen' are very similar words in English, perhaps as a result of the long reigns of the two Queens Elisabeth and Queen Victoria.

The hypothetical gender vector, which should solve the `family` analogies, does not occur between 'King' and 'Queen'. 'Man' and 'woman' are also nearest neighbors, so the computed neighborhood of the answer is near 'King'; 'Queen' turns out to be the nearest word except for 'King'.

5 Discussion and Further Work

We have found a relation-specific linear transformation which improves the evaluation of word-analogies with vector arithmetic, including those pairs in the same relation which were held out of the training set. We believe that we are the first to consider this particular approach. We think that the approach works because the trained transformation discards information which is not common to the various pairs, while amplifying information which is. However, except for analogy performance, we have no evidence of this.

It seems possible that the analogy relations could be refined so that more of the pairs actually showed the same relationship; one way to do this might be to track the performance of individual pairs; those which consistently perform poorly in the held-out set may be candidate for removal. If word analogies can be enhanced to be more reliable, as our work suggests, then they may be good tools for obtaining semantic, morphological, or lexical information from semantic spaces. An advantage of linear transformations should be that it is easier to interpret how they interact with the data, compared to e.g. neural networks.

It might be that psychological analysis of meaning axes, as considered by Hollis and Westbury [5] could make use of this tool. A number of the parameters of this experiment were chosen arbitrarily. For example, refraining from normalization, while plausible, may not be necessary, and normalization may in fact have the salutary effects reported in other work. Other strategies to keep the vector space from condensing may work better than negative examples. The parameter space deserves to be explored more fully.

Acknowledgments. This work has been supported by the project LO1506 of the Czech Ministry of Education, Youth and Sports. Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated.

References

1. Coecke, B., Sadrzadeh, M., Clark, S.: Mathematical foundations for a compositional distributional model of meaning. Lambek Festschrit, special issue of Linguistic Analysis, vol. 36, pp. 345–384 (2010)
2. Drozd, A., Gladkova, A., Matsuoka, S.: Word embeddings, analogies, and machine learning: Beyond king-man + woman = queen. In: COLING (2016)
3. Finley, G. P., Farmer, S., Pakhomov, S. V.: What analogies reveal about word vectors and their compositionality. Proceedings of the 6th Joint Conference on Lexical and Computational Semantics, (2017)
4. Gittens, A., Achlioptas, D., Mahoney, M. W.: Skip-gram – zipf + uniform = vector additivity. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 69–76 (2017)
5. Hollis, G., Westbury, C. F.: The principals of meaning: Extracting semantic dimensions from co-occurance models of semantics. Psychonomic Bulletin and Review, vol. 23 (2016)

6. Konkol, M., Brychcín, T., Nykl, M., Hercig, T.: Geographical evaluation of word embeddings. In: Proceedings of the The 8th International Joint Conference on Natural Language Processing. pp. 224–232 (2017)
7. Levy, O., Goldberg, Y.: Linguistic regularities in sparse and explicit word representations. Proceedings of the Eighteenth Conference on Computational Language Learning, pp. 171–180 (2014)
8. Levy, O., Remusu, S., Biemann, C., Dagan, I.: Do supervised distributional methods really learn lexical inference relations? In: North American Chapter of the Association for Computational Linguistics Human Language Technologies (NAACL HLT 2015) (2015)
9. Linzen, T.: Issues in evaluating semantic spaces using word analogies. Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP. Association for Computational Linguistics, (2016)
10. Manning, C. D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
11. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. Proceedings of the International Conference on Learning Representations, (2013)
12. Mikolov, T., Sutskever, I., Chen, K., Dean, J.: Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems, (2013)
13. Mikolov, T., Tau Yih, W., Zweig, G.: Linguistic regularities in continuous space word representations. HLT-NAACL, vol. 13, pp. 746–751 (2013)
14. Mitchell, J., Lapata, M.: Composition in distributional models of semantics. Cognitive Science, vol. 34, pp. 1388–1429 (2010)
15. Shusen, L., Peer, T. B., Jayaraman, J. T., Vivek, S., Bei, W., Yarden, L., Valerio, P.: Visual exploration of semantic relationships in neural word embeddings. Transactions on Visualization and Computer Graphics, vol. 24, no. 1, pp. 553–562 (2018)
16. Szymanski, T.: Temporal word analogies: Identifying lexical replacement with diachronic word embeddings. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers). pp. 448–453 (2017)
17. Turney, P. D., Littman, M. L.: Corpus-based learning of analogies and semantic relations. Machine Learning, vol. 60, no. 1, pp. 251–278 (2005)
18. Vylomova, E., Rimell, L., Cohn, T., Baldwin, T.: Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. pp. 1671–1682 (2016)

Electronic edition
Available online: http://www.rcs.cic.ipn.mx

ISSN: 1870-4069
http://rcs.cic.ipn.mx

Centro de Investigación
en Computación