ISSN 1870-4069

Optimización evolutiva de contextos para la corrección fonética en sistemas de reconocimiento del habla

Rafael Viana Cámara, Diego Campos Sobrino, Mario Campos Soberanis

SoldAI Research, Mérida, México
{rviana,dcampos,mcampos}@soldai.com

Resumen El reconocimiento automático del habla es un área de creciente interés por la demanda de aplicaciones que lo utilizan para brindar una forma de comunicación natural a los usuarios. Es común que los sistemas de reconocimiento presenten fallas en aplicaciones que usan un lenguaje propio de un dominio específico. Para reducir el error se utilizan diferentes estrategias como proporcionar un contexto que modifique el modelo de lenguaje y un pos-procesamiento de corrección. En este artículo se explora el uso de un proceso evolutivo para la generación de un contexto optimizado al dominio de aplicación, así como diferentes técnicas de corrección basadas en métricas de distancia fonética. Los resultados obtenidos muestran la viabilidad de los algoritmos genéticos como una herramienta de optimización de los contextos, lo cual sumado a un pos-procesamiento de corrección basado en representaciones fonéticas es capaz de reducir los errores en el reconocimiento.

Palabras clave: reconocimiento del habla, distancia fonética, algoritmos genéticos.

Evolutionary Optimization of Contexts for Phonetic Correction in Speech Recognition Systems

Abstract. Automatic Speech Recognition (ASR) is an area of growing academic and commercial interest due to the high demand for applications that use it to provide a natural way of communication. It is common for general purpose ASR systems to fail in certain applications that use a domain specific language. Different strategies have been used to reduce the error, such as providing a context that modifies the language model and post-processing correction methods. This article explores the use of an evolutionary process to generate an optimized context for a specific application domain, as well as different correction techniques based on phonetic distance metrics. The results show the viability of a genetic algorithm as a tool for context optimization, which added to a post-processing correction based on phonetic representations is able to reduce the errors on the recognized speech.

Keywords: speech recognition, phonetic distance, genetic algorithms.

1. Introducción

Los sistemas de reconocimiento automático del habla (ASR por sus siglas en inglés) resultan de gran relevancia en entornos académicos y empresariales debido a la facilidad de interacción que ofrecen. Se ha visto un creciente interés en la investigación de dichos sistemas los cuales han migrado de modelos probabilísticos a sistemas de redes neuronales profundas [5] que se han convertido en el estándar para aplicaciones profesionales de transformación de audio a texto. Estos sistemas a menudo utilizan un modelo acústico para realizar el reconocimiento en un primer nivel y posteriormente son pasados a modelos de lenguaje para su corrección [9]. Los servicios comerciales generalmente funcionan como una caja negra, imposibilitando al usuario la modificación de los modelos de lenguaje.

Si bien los ASR tienen un buen rendimiento de manera general, con frecuencia se enfrentan con problemas al utilizarlos para reconocer dominios específicos de lenguaje por lo que las técnicas de pos-procesamiento cobran relevancia [4].

Muchas de las tareas de pos-procesamiento y corrección de estos sistemas utilizan un contexto, entendido como un conjunto de palabras, frases y expresiones relacionadas al dominio particular que se desea reconocer. Algunos de ellos cuentan con mecanismos para proporcionar un contexto con el cual mejoran el reconocimiento de ciertas palabras y frases, sin embargo, en muchas ocasiones no es suficiente para mejorar de manera significativa su desempeño.

Dos temas particularmente interesantes son la generación de contextos y la representación fonética para la corrección. Se ha realizado investigación al respecto, sin embargo, falta experimentación relacionada al funcionamiento conjunto de la generación del contexto y representación fonética óptimas.

En este artículo se presenta un método para la generación de contextos utilizando algoritmos genéticos para corregir la salida del sistema de procesamiento de voz a texto de *Google*. A continuación se realiza la comparación de diferentes estrategias críticas en el proceso de corrección de errores: representación de la frase a corregir, selección de candidatos y métricas de comparación.

El artículo está estructurado de la siguiente manera: en la sección 2 se describen los antecedentes del problema y trabajos relacionados; la sección 3 presenta la metodología empleada para la investigación; en la sección 4 se describe el trabajo experimental realizado cuyos resultados se muestran en la sección 5 y finalmente en la sección 6 se proporcionan las conclusiones junto con algunas ideas para desarrollar como trabajo futuro.

2. Antecedentes

Los algoritmos de corrección de errores en sistemas ASR se han abordado desde diferentes perspectivas, incluyendo las fonéticas. Kondrak [11] plantea un algoritmo para calcular una métrica de similitud fonética entre segmentos utilizando características fonéticas articulatorias multivalor. El algoritmo de Kondrak combina conjuntos de operaciones de edición y modelos de alineación

locales y semiglobales para calcular un conjunto de alineamientos casi óptimos.

Pucher et al. [16] presentan matrices de confusión de palabras mediante el uso de diferentes medidas de distancia fonética. Las métricas presentadas se basan en la distancia de edición mínima entre las transcripciones fonéticas y las distancias entre modelos ocultos de Markov. Su investigación muestra que existe una correlación entre la distancia de edición y la confusión de las palabras en sistemas ASR, por lo que este tipo de correcciones se vuelven útiles para rectificar errores de reconocimiento.

En [2] se resalta la problemática que representa usar la distancia de edición para comparar cadenas en lenguajes como el coreano, donde los caracteres representan sílabas en lugar de letras. Lo anterior se refleja en el hecho de que sustituir una sílaba por otra aporta el mismo valor independientemente de la diferencia entre sus letras. La solución tradicional utiliza métricas híbridas entre caracteres y sílabas, sin embargo, los autores argumentan que este enfoque no resuelve satisfactoriamente el problema por lo que proponen una distancia de edición basada en fonemas como solución.

Droppo y Acero [9] utilizan la distancia de edición fonética para incorporar un tercer elemento de corrección a los sistemas ASR. Los investigadores incorporan esta distancia para aprender la probabilidad relativa de cadenas de reconocimiento fonético, dada una pronunciación esperada. Esta estrategia toma en cuenta el contexto de las transcripciones, cambiando la probabilidad de la corrección dependiendo de las palabras previas y posteriores.

Bassil y Semaan [4] emplean una estrategia de pos-procesamiento para la corrección de errores en sistemas ASR. El método presentado para detectar errores en las palabras utiliza un algoritmo de generación de candidatos y uno de corrección de errores sensible al contexto. Los autores reportan una reducción importante de los errores del sistema.

En [7] se emplean estrategias de corrección fonética para corregir los errores generados por un sistema ASR. En el trabajo citado se pasa la transcripción devuelta por el sistema a una representación en formato de Alfabeto Fonético Internacional (IPA por sus siglas en inglés) y un algoritmo de ventana deslizante para la selección de frases candidatas para su corrección de acuerdo a las palabras provistas en el contexto y la distancia a su representación fonética en formato IPA. Los autores reportan una mejora en el 30 % de las frases reconocidas por el servicio de *Google*.

Un componente importante para el algoritmo de corrección fonética es el contexto utilizado para construir las frases candidatas, por lo que se necesitan soluciones capaces de buscar configuraciones óptimas entre espacios de búsqueda extremadamente grandes.

Los algoritmos genéticos son algoritmos de búsqueda estocástica basados en los principios de evolución biológica y emulan el proceso por medio de operadores genéticos aplicando recombinación, mutación y selección natural en una población [13,18]. Se han aplicado para resolver problemas combinatorios complejos y los resultados muestran que constituyen una estrategia potente y eficiente cuando son utilizados de manera correcta. [18].

Este tipo de algoritmos se han utilizado para analizar una gran cantidad de problemas entre los que se encuentran: knapsack [18], problemas de calendarización de procesos, el vendedor viajero [13], búsqueda de funciones para regresión simbólica [1], funciones de kernel gaussiano para análisis de sentimientos [17], entre otros.

Para utilizar algoritmos genéticos se expresan las posibles soluciones como una cadena de símbolos llamada cromosoma, donde cada símbolo es un gen. A partir de una generación inicial de individuos, se ejecutan iterativamente los procesos de selección, mutación, recombinación y evaluación, combinando los genes de los individuos para producir nuevas variaciones. Cada individuo es evaluado de acuerdo a una función llamada *fitness* que describe que tan bien se desempeña como una solución al problema que se intenta resolver.

Los algoritmos genéticos resultan efectivos para explorar espacios amplios de búsqueda, sin embargo, para ciertas configuraciones y problemas, el tiempo de procesamiento necesario para encontrar una solución óptima puede incrementar considerablemente, por lo que a menudo se utilizan técnicas para variar el tamaño de la población y el rango de mutación [13].

En esta investigación se decidió utilizar algoritmos genéticos para la optimización del contexto debido a la efectividad que ha sido reportada en los diferentes dominios de aplicación en los cuales se han utilizado, así como por su capacidad de explorar una gran parte del espacio de búsqueda.

El presente artículo extiende la investigación previa de [7] tomando en cuenta la generación automática de contextos mediante estrategias evolutivas para mejorar la transcripción y la corrección de errores de sistemas ASR utilizando representaciones fonéticas. Se extiende el algoritmo original para comparar los resultados entre diferentes representaciones fonéticas, estrategias para la selección de candidatos de corrección y métricas de distancia. Se utilizan algoritmos genéticos para optimizar el contexto pasado a los algoritmos de corrección para mejorar el reconocimiento, utilizando una función de *fitness* que evalúa el número de palabras corregidas por un contexto determinado.

3. Metodología

El algoritmo de corrección utiliza los componentes de un contexto (conjunto de palabras y frases propias del dominio de la aplicación) para detectar posibles errores en el reconocimiento y corregir la transcripción de un sistema de reconocimiento automático del habla. Está compuesto por tres elementos principales: representación fonética, generador de frases candidatas para corrección y métrica de distancia de edición. Como medida de evaluación de los resultados se utilizó la métrica WER (*Word Error Rate*), que se define como sigue:

$$WER = \frac{S + D + I}{N}, \tag{1}$$

donde S es el número de sustituciones, D el número de supresiones, I el número de inserciones necesarias para transformar la frase hipotética en la frase real y N el número de palabras en la frase real.

3.1. Transcripción fonética

La transcripción fonética es un sistema de símbolos gráficos que representan los sonidos del habla humana. Es usado como convención para evitar las peculiaridades de cada lengua escrita y representar aquellas lenguas sin tradición escrita [10]. Utilizamos como representaciones fonéticas: texto simple, IPA, Double Metaphone (DM) y una variante del Double Metaphone con vocales (DMV).

El IPA es un sistema de notación fonética basado en el alfabeto latino, utilizado como una representación estandarizada de los sonidos del lenguaje hablado [6,19]. Metaphone es un algoritmo fonético que se encarga de indexar palabras por su pronunciación con origen en el idioma inglés [15]. El algoritmo DM es una versión mejorada del algoritmo Metaphone, la cual devuelve una representación del sonido de las letras en la cadena cuando el texto es pronunciado y omite las vocales. El DM ha sido utilizado a menudo para representación del idioma inglés, sin embargo, los sonidos de las vocales resultan de importancia en el español debido a que sirven al hispanohablante para enlazar palabras que terminan en grupos consonánticos [8], por lo que desarrollamos una variante del DM la cual añade las vocales que son eliminadas en el algoritmo original.

3.2. Algoritmos de generación de frases candidatas

Durante el proceso de corrección fonética la búsqueda de frases candidatas genera segmentos de la cadena de entrada que serán contrastadas mediante una métrica de distancia con las palabras y frases del contexto. Una frase candidata es aquella que tiene similitud con alguna de las frases del contexto y que puede contener un error en la transcripción del sistema ASR. Como algoritmos de generación de frases candidatas se utilizaron el de ventana pivotal y el de comparación incremental basado en el tamaño de la frase en letras o sílabas.

En [7] se presenta la estrategia de ventana deslizante en donde se genera un conjunto S_j con una ventana $v = 1$. La selección de subfrases de realiza utilizando un pivote p_j y el conjunto S_j de frases candidatas se genera por las subfrases $\{p_j,\ p_{j-1}p_j,\ p_jp_{j+1},\ p_{j-1}p_jp_{j+1}\}$.

Para este artículo se implementó un método de búsqueda incremental de subfrases que se describe a continuación:

Sea $C = \{c_1, \ldots, c_n\}$ el conjunto de n frases específicas del contexto, $T = \{t_1, \ldots, t_m\}$ la transcripción original dividida en m palabras, se pretende construir un conjunto $R = \{(s_1, c_1), \ldots, (s_l, c_l)\}$ formado por pares (s_i, c_i) tales que c_i es un elemento del contexto susceptible de sustituir el segmento $s_i = t_j...t_k$ para algún $j \leq k$ en T.

El algoritmo 1 muestra la estrategia utilizada, en la que a partir de cada palabra t_i de la transcripción T se incrementa palabra por palabra la subfrase s a evaluar, hasta que ya no queden elementos del contexto comparables por su tamaño en letras o sílabas de acuerdo al umbral u. Las posibles sustituciones de s por c son agregadas al conjunto R siempre y cuando su distancia sea menor a u. La complejidad del algoritmo es $O(nm^2)$ donde n es el número de elementos del contexto y m el tamaño de la transcripción en palabras.

Algoritmo 1 Algoritmo de búsqueda incremental de candidatos.

Input: El contexto $C = \{c_1, \ldots, c_n\}$, la transcripción $T = \{t_1, \ldots, t_m\}$, un umbral de distancia u, una función de distancia de edición $d(a, b)$.

Output: un conjunto $R = \{(s_1, c_1), \ldots, (s_l, c_l)\}$ de sustituciones candidatas.

```
1: Calcular el tamaño máximo de frase L_M en C.
2: Inicializa el conjunto R = {}
3: for i ← 1...m do
4:     s ← t_i
5:     j ← i
6:     while j ≤ m & length(s) ≤ L_M/(1−u) do
7:         for all c ∈ C | length(s)(1−u) ≤ length(c) ≤ length(s)/(1−u) do
8:             if d(s, c) < u then
9:                 Añade el par (s, c) al conjunto R
10:            end if
11:            s ← s + t_j
12:            j ← j + 1
13:        end for
14:    end while
15: end for
16: return R
```

3.3. Métricas de distancia de cadenas

La distancia de edición es utilizadas para cuantificar la diferencia entre dos cadenas de texto en términos del número de operaciones necesarias para transformar una cadena en la otra. En este artículo se experimenta con las métricas de distancia de Levenshtein, Damerau-Levenshtein y Alineación Óptima de Cadenas (OSA por su siglas en inglés).

La distancia de Levenshtein entre dos cadenas de caracteres es el número de inserciones, eliminaciones y sustituciones necesarias para transformar una cadena de caracteres en otra [12]. La distancia Damerau-Levenshtein se puede definir intuitivamente como una extensión de la distancia de Levenshtein añadiendo como operación válida la transposición de dos caracteres adyacentes [3]. OSA es una variación restrictiva de la distancia *Damerau-Levenshtein*, en donde la operación de transposición solo puede ser efectuada una vez por caracter [14], lo cual la hace menos costosa computacionalmente.

3.4. Optimización evolutiva del contexto

Para la generación de contextos se decidió utilizar un algoritmo genético construido a partir de las transcripciones de las frases a corregir. Para la construcción de los individuos se consideraron todas las palabras individualmente, así como las combinaciones de 2 palabras (bigramas) presentes en las frases objetivo de los audios originales. Los individuos fueron definidos por un cromosoma donde cada gen toma el valor 1 si la palabra o bigrama se encuentra en el contexto y 0 en caso contrario.

Algoritmo 2 Algoritmo de optimización del contexto.

Input: Tamaño de población N, número de generaciones G, tamaño del torneo T_s, probabilidad de cruce C_p y probabilidad de mutación M_p.

Output: La población evolucionada p y el error promedio por cada generación $errores$.

1: $p \leftarrow GeneraPoblacionInicial(N)$, $g \leftarrow 0$
2: **while** $g < N$ **do**
3: $\quad errores[g] \leftarrow Evaluar(p)$
4: $\quad p_t \leftarrow Seleccionar(p, T_s)$
5: $\quad p \leftarrow CruzarMutar(p_t, C_p, M_p)$
6: $\quad g \leftarrow g + 1$
7: **end while**
8: **return** p, $errores$

De esta manera cada individuo representa un contexto que puede ser dado como parámetro al algoritmo de corrección. Para evaluar a los individuos, se ejecutó el algoritmo de corrección con la mejor combinación encontrada en el artículo [7] a cada una de las 451 frases y se devolvió como medida de *fitness* el WER total de cada contexto analizado. Se utilizó un algoritmo genético simple descrito en el algoritmo 2 donde la función $GeneraPoblacionInicial(N)$ produce N individuos de manera aleatoria, $Evaluar(p)$ es una función que calcula el WER de cada individuo, lo asigna como medida de *fitness* y devuelve el WER promedio de la población. La selección se realizó con una estrategia simple de torneo.

La función $CruzarMutar(p_t, C_p, M_p)$ realiza la recombinación genética entre los individuos de la población utilizando la técnica de punto de cruce aleatorio mostrada en el algoritmo 3. Posteriormente se realizó el proceso de mutación individuo a individuo y gen a gen de acuerdo al valor de la probabilidad de mutación, la cual se fue reduciendo cada 10 generaciones para disminuir la fluctuación y estabilizar el error cuando se acerca a un mínimo.

Algoritmo 3 Operación de cruce entre individuos.

Input: Individuos I_1 e I_2 a combinarse, un punto de cruce c_i, y un tamaño de cromosoma c_{size}

Output: Descendencia de los individuos de entrada definida como H_1 y H_2.

1: $H_1 \leftarrow g1_1 g1_2 ... g1_{c_i} g2_{c_i+1} g2_{c_i+2} ... g2_{c_{size}}$
2: $H_2 \leftarrow g2_1 g2_2 ... g2_{c_i} g1_{c_i+1} g1_{c_i+2} ... g1_{c_{size}}$
3: **return** H_1, H_2

4. Experimentación

La experimentación en el presente trabajo se realizó utilizando 451 frases transcritas por el sistema de reconocimiento del habla de *Google*. Este mismo

corpus fue utilizado en [7], donde también se pueden obtener detalles del proceso de recolección y formato de dicho corpus.

4.1. Variantes de configuración del corrector

Con el propósito de comparar las variantes de cada uno de los elementos del algoritmo se realizaron un total de 72 experimentos con todas las combinaciones de los métodos presentados en la tabla 1.

Tabla 1. Variantes de métodos para cada elemento del algoritmo.

Representación	Generación frases	Métrica distancia	Google STT
Texto simple	WIN	Levenshtein	Básico
IPA	LET	OSA	Contextual
DM	SYL	Damerau-Levenshtein	
DMV			

El texto de la frase reconocida por el sistema STT y las frases del contexto fueron representadas de diferente manera para su procesamiento. Para todas las representaciones de texto fue necesario realizar un proceso de normalización con el fin de remover símbolos y caracteres extraños, luego esta versión normalizada fue utilizada directamente en forma de texto simple o transformada a una de las representaciones fonéticas analizadas: Alfabeto Fonético Internacional (IPA), Double Metaphone (DM) o Double Metaphone con vocales (DMV).

Se experimentó con la generación de frases candidatas usando los métodos de ventana pivote (WIN) y la comparación incremental de acuerdo al número de caracteres (LET) o sílabas (SYL).

Como métricas de distancia de edición fueron usadas la distancia de Levenshtein y sus variantes OSA y Damerau-Levenshtein. Cada combinación fue probada usando como datos de entrada las 451 transcripciones obtenidas al usar el método básico de *Google* y posteriormente la transcripción resultante de enviar el contexto reportado en [7] al servicio de *Google*.

Para cada una de las 72 diferentes configuraciones se experimentó con diferentes umbrales de coincidencia de las métricas de edición con incrementos de 0.05 hasta un máximo de 0.6. La evaluación de cada configuración experimental se realizó utilizando la métrica WER acumulada globalmente como resultado de calcular el número de ediciones necesarias para transformar la transcripción hipotética en la frase correcta para cada uno de los 451 ejemplos.

4.2. Optimización del contexto

La experimentación descrita en la sección anterior fue realizada con un contexto generado empíricamente de acuerdo al conocimiento a priori de las frases del dominio en donde se observaban fallos en la transcripción. Con el fin de

optimizar el contexto se ejecutó un algoritmo genético cuyos parámetros fueron calibrados realizando una experimentación de 30 ejecuciones con una versión reducida del problema usando un cromosoma de tamaño 50, arrojando los mejores resultados con una población de 50 individuos, 100 generaciones, 95 % de probabilidad de cruce y 5 % de probabilidad de mutación.

Una vez calibrados los parámetros se optimizó el contexto usando un tamaño de cromosoma de 355 donde cada gen representa una de las palabras o bigramas presentes en las transcripciones de los audios de la experimentación. El factor de mutación se fue reduciendo en un 20 % cada 10 generaciones. Los individuos estuvieron evolucionando durante 100 generaciones.

Como función de *fitness* se empleó el WER total obtenido al ejecutar el algoritmo de corrección para la transcripción simple de *Google* utilizando como contexto al individuo definido por el cromosoma. El corrector fonético se ejecutó utilizando representación IPA, selección de ventana pivote, distancia de Levenshtein y umbral de 0.4, la cual fue la mejor configuración reportada en [7].

Se realizaron 5 procesos evolutivos de 100 generaciones donde se inicializó la población con los 25 mejores individuos de la ronda anterior y 25 generados al azar, para poder explorar diferentes variantes evolutivas. La experimentación se realizó con un procesador intel i7, 8GB de RAM, un sistema operativo Debian GNU/Linux. El algoritmo fue implementado en Python3 y se ejecutó 5 veces durante un total de 70 horas.

4.3. Corrección de errores con el contexto optimizado

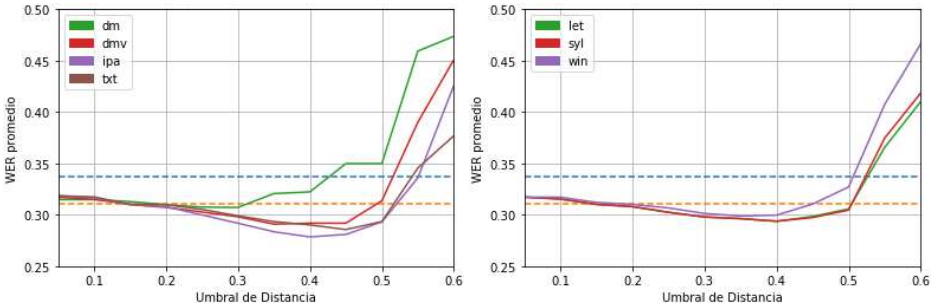
En esta fase de la experimentación se realizaron pruebas para medir los efectos del contexto en el reconocimiento del habla y en el proceso de corrección posterior. Se utilizaron los archivos de audio previamente generados, mismos que fueron enviados de nuevo al sistema de reconocimiento en su modalidad básica y con el contexto genéticamente generado. Esto nos dio una comparación directa sobre el efecto que tiene usar el contexto optimizado con relación a la transcripción obtenida sin enviar contexto, además, nos proporcionó dos lineas de base sobre las cuales aplicar el proceso de corrección a las nuevas transcripciones.

Para comparar los resultados del proceso de corrección, el algoritmo fue aplicado a las nuevas transcripciones producidas por ambas versiones del reconocedor de *Google*. Se usaron las cuatro variantes de configuración que presentaron mejores resultados en la experimentación descrita en la sección 4.1. Tomando como entrada las transcripciones obtenidas por las dos modalidades del servicio de *Google* se ejecutaron dos experimentos para cada una de ellas. En el primero se usó el contexto experimental utilizado en [7], el cual denotaremos como C_m y en el segundo experimento el nuevo contexto generado genéticamente C_g.

5. Resultados

La Fig. 1(a) muestra la variación en el WER promedio obtenido en los experimentos agrupados por el modo de representación con diferentes umbrales

de distancia. Las líneas horizontales corresponden al WER obtenido con la transcripción del STT básico (33.7 %) y el STT contextual (31.1 %). Se observa el efecto de reducción en el WER que se tiene al transformar el texto simple en IPA, sobre todo alrededor del valor 0.4 para el umbral, donde se alcanza un WER promedio mínimo de 27.8 %. Por su parte, la representación en DM produjo resultados similares a los demás métodos para valores pequeños del umbral, sin embargo alrededor de 0.3 empieza a decrecer su capacidad correctiva de manera consistente. La versión de DMV sostiene su rendimiento a la par que el texto simple hasta un umbral de 0.4.



(a) Representación de texto (b) Generación de candidatos

Fig. 1. WER promedio para diferentes representaciones de texto (a) y métodos de generación de candidatos (b).

En la Fig. 1(b) se observa el WER promedio obtenido por las diferentes configuraciones experimentales agrupadas por el algoritmo de generación de frases candidatas. La gráfica muestra un mejor desempeño de las variantes de comparación incremental, ya sea por el tamaño en letras o en sílabas, en relación con la ventana pivote. El WER promedio mínimo (29.4 %) se alcanza con un umbral de 0.4 para el método LET. La versión SYL muestra resultados muy similares, sin embargo el costo computacional de su procesamiento es mayor.

Los mejores resultados se obtuvieron con la configuración de representación IPA y método de selección de candidatos LET. El WER obtenido a partir de la transcripción básica se redujo de 33.7 % a 28.1 % y para la transcripción contextual se redujo de 31.1 % a 27.3 %. Esta configuración presentó una reducción global en el WER relativo de 19.3 %. En relación con las tres métricas de distancia evaluadas la diferencia en los resultados fue prácticamente nula.

Los resultados obtenidos a partir de la experimentación con los algoritmos genéticos para la optimización del contexto utilizando la configuración experimental descrita en la sección 4.2, indican que se obtuvo un error promedio en la quinta ejecución del experimento de 26.5 %, el cual comenzó con un error promedio de 31.2 % y se redujo hasta un promedio de 24.9 % en la generación 100.

En la Fig. 2 se muestran los errores promedio de las 100 generaciones en la quinta ejecución del experimento. El mejor contexto encontrado con esta estrategia quedó conformado por 64 unigramas y 117 bigramas con un WER total de 24.7. %.

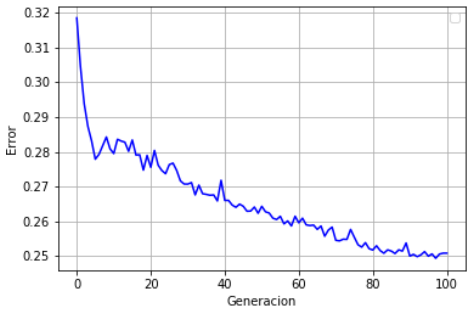
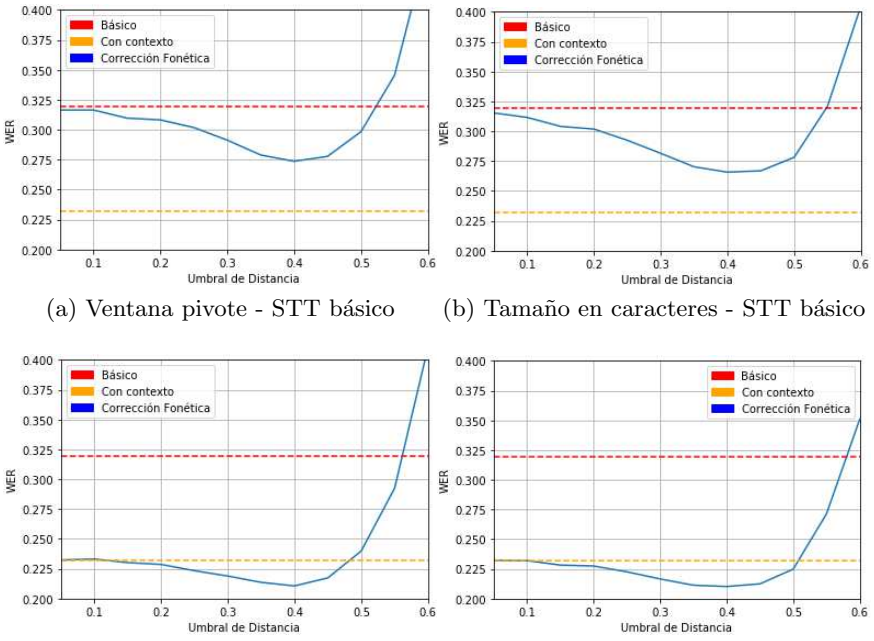


Fig. 2. Gráfica del error promedio por generación del algoritmo genético.



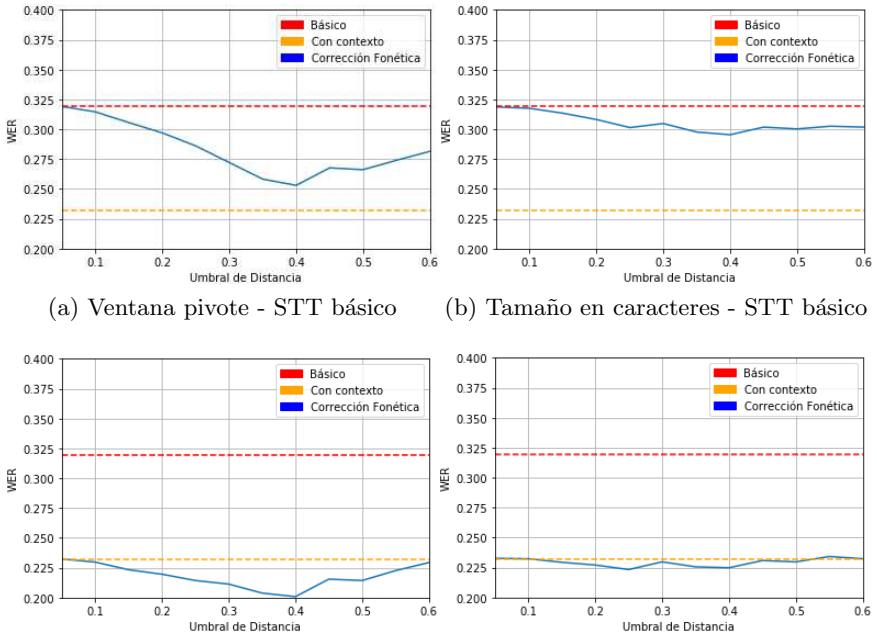
(a) Ventana pivote - STT básico (b) Tamaño en caracteres - STT básico

(c) Ventana pivote - STT contextual (d) Tamaño en caracteres - STT contextual

Fig. 3. Resultados usando C_m como entrada del algoritmo de corrección.

En la fase final de la experimentación descrita en la sección 4.3 se obtienen dos lineas de base a partir de las cuales ejecutar el proceso de corrección. El WER obtenido al comparar las frases reales pronunciadas con la transcripción básica fue de 32.0 %, mientras que al incorporar el contexto generado genéticamente el WER se redujo considerablemente hasta un 23.2 %. Este resultado nos permite ver el impacto que tiene un contexto optimizado en el modelo de lenguaje usado por *Google* al reducir el WER relativo en un 27.3 %.

En la Fig. 3 se ven los resultados del algoritmo de corrección usando la representación IPA con el contexto C_m. Partiendo del STT básico el WER mínimo se obtiene con un umbral de 0.4 y el proceso de selección LET, con esa configuración el WER total se reduce de 32.0 % a 26.6 %, lo que representa una reducción de 16.9 % en el WER relativo. Al iniciar el proceso de corrección a partir de la transcripción del STT contextual el WER se reduce de 23.2 % a 21.0 %, alcanzando una mejora de 9.5 % en el WER relativo.



(a) Ventana pivote - STT básico (b) Tamaño en caracteres - STT básico

(c) Ventana pivote - STT contextual (d) Tamaño en caracteres - STT contextual

Fig. 4. Resultados usando C_g como entrada del algoritmo de corrección.

De manera similar, la Fig. 4 muestra resultados usando como entrada al algoritmo de corrección el contexto generado genéticamente C_g. El uso de este contexto permite reducir el WER mínimo cuando se usa el procedimiento de generación de candidatos WIN, pero parece no generar muy buenos resultados

con el método LET. El mínimo WER absoluto para el STT básico es de 25.3 % con lo que se alcanza una reducción en WER relativo del 21.0 %. Partiendo del STT contextual se alcanza un mínimo de 20.1 % lo que representa una reducción del 13.6 % en WER relativo.

6. Conclusiones y trabajo futuro

A partir de los resultados obtenidos en la experimentación, se muestra la utilidad del algoritmo de corrección fonética para reducir los errores de la transcripción de *Google*, tanto en su versión básica como en la contextual. Se observa que la mejor configuración para el algoritmo se obtiene utilizando IPA como representación fonética y la selección incremental por letras, logrando reducir el WER relativo en un 19.0 %.

De igual manera podemos mencionar que los algoritmos genéticos constituyen una alternativa eficaz para la generación de contextos, puesto que consiguió reducir el WER de la transcripción básica de *Google* de un 32.0 % a 23.2 %. El contexto mostró tener un valor crucial en el desempeño del algoritmo.

Los mejores resultados se obtuvieron de la combinación de la corrección fonética con la optimización evolutiva del contexto logrando una reducción del WER absoluto de 11.9 % al disminuirlo de 32.0 % a 20.1 %, lo que representa una mejora en WER relativo de 37.2 %.

El hecho de que tanto el algoritmo de corrección fonética como la optimización evolutiva del contexto sean independientes del sistema usado para la transcripción y del dominio de aplicación hace que la estrategia presentada pueda ser extendida a diferentes sistemas ASR y dominios de aplicación.

Los algoritmos presentados a lo largo de este artículo pueden tomar ventaja del conocimiento a priori del dominio de aplicación para mitigar el problema del comienzo en frío. Lo anterior se debe a que en caso de que no se cuenten con las transcripciones iniciales, se puede utilizar un contexto generado con el conocimiento humano del dominio, como en [7], el cual puede complementarse con los algoritmos genéticos a medida que se recopila información acerca de las interacciones con usuarios reales del sistema.

Entre las líneas de investigación a futuro se requiere validar los resultados con corpus de diferentes dominios de aplicación, además se prevé la experimentación utilizando costos de edición ponderados tomando en cuenta características fonéticas del español y del audio original tales como ruido, duración, energía de la señal, entre otras. Otra línea de investigación es la comparación con algoritmos de *deep learning*, ya que se puede plantear el problema de la corrección de errores en sistemas ASR como una traducción de transcripciones erróneas a transcripciones correctas, por lo que algoritmos de *Machine translation* pueden resultar de utilidad.

Referencias

1. Anjum, A., Sun, F., Wang, L., Orchard, J.: A novel continuous representation of genetic programmings using recurrent neural networks for symbolic regression. CoRR abs/1904.03368 (2019), http://arxiv.org/abs/1904.03368
2. Bae, B., Kang, S.s., Hwang, B.y.: Edit Distance Calculation by Phonetic Rules and Word-length Normalization 2 Related Works 3 Edit Distance for Korean Words 4 Phoneme-based Edit Distance. Advances in Computer Science (1), 315–319 (2012)
3. Bard, G.V.: Spelling-error tolerant, order-independent pass-phrases via the damerau-levenshtein string-edit distance metric. In: Proceedings of the Fifth Australasian Symposium on ACSW Frontiers - Volume 68. pp. 117–124. ACSW '07, Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2007), http://dl.acm.org/citation.cfm?id=1274531.1274545
4. Bassil, Y., Semaan, P.: ASR context-sensitive error correction based on Microsoft n-gram dataset. CoRR abs/1203.5262 (2012)
5. Becerra, A., de la Rosa, J.I., González, E.: A case study of speech recognition in Spanish: From conventional to deep approach. In: 2016 IEEE ANDESCON. pp. 1–4 (Oct 2016)
6. C., M.M.K.: Phonetic notation. The World's Writing Systems pp. 821–846 (1996)
7. Campos Sobrino, D., Campos Soberanis, M.A., Martínez Chin, I., Uc Cetina, V.: Correccion de errores del reconocedor de voz de google usando métricas de distancia fonética. Research in Computing Science (In Press) (04 2018)
8. Chela-Flores, B.: Consideraciones teórico-metodológias sobre la adqusición de consonantes posnucleares del inglés. RLA. Revista de lingüística teórica y aplicada 44 (12 2006)
9. Droppo, J., Acero, A.: Context dependent phonetic string edit distance for automatic speech recognition. In: 2010 IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 4358–4361 (March 2010)
10. Hualde, J.: The sounds of Spanish. Cambridge University Press (2005)
11. Kondrak, G.: Phonetic alignment and similarity. Computers and the Humanities 37(3), 273–291 (Aug 2003), https://doi.org/10.1023/A:1025071200644
12. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics Doklady 10(8), 707–710 (feb 1966), doklady Akademii Nauk SSSR, V 163, No 4, pp. 845–848 (1965)
13. Luo, J., Baz, D.E.: A survey on parallel genetic algorithms for shop scheduling problems. CoRR abs/1904.04031 (2019), http://arxiv.org/abs/1904.04031
14. Navarro, G.: A guided tour to approximate string matching. ACM Comput. Surv. 33(1), 31–88 (Mar 2001), http://doi.acm.org/10.1145/375360.375365
15. Philips, L.: Hanging on the metaphone. Computer Language Magazine 7(12), 39–44 (December 1990), accessible at http://www.cuj.com/documents/s=8038/cuj0006philips/
16. Pucher, M., Turk, A., J., A., Fecher, N.: Distance, phonetic and for, measures and recognition, speech and optimization, grammar. 3rd Congress of the Alps Adria Acoustics Association (2007)
17. Roman, I., Mendiburu, A., Santana, R., Lozano, J.A.: Sentiment analysis with genetically evolved Gaussian kernels. CoRR abs/1904.00977 (2019), http://arxiv.org/abs/1904.00977
18. Shah, S.: Genetic algorithm for a class of Knapsack problems. CoRR abs/1903.03494 (2019), http://arxiv.org/abs/1903.03494
19. Wall, J.: International phonetic alphabet for singers: A manual for English and foreign language diction. Dallas, Tex.: Pst (1989)