PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

IMPLEMENTACIÓN DE UN CORRECTOR ORTOGRÁFICO PARA LENGUAS ORIGINARIAS DEL PERÚ. CASO DE ESTUDIO: SHIPIBO-KONIBO

TESIS PARA OPTAR EL TÍTULO PROFESIONAL EN INGENIERíA INFORMATICA

AUTOR

Carlo André Alva Cohello

ASESOR:

Mag. Félix Arturo Oncevay Marcos

Lima, diciembre del 2018

Resumen

En el Perú existen diversas lenguas originarias como el shipibo-konibo, asháninka, el kakataibo, entre otras [Rivera, 2001]. Estas lenguas se caracterizan porque son transmitidas a través de cuentos, poesía y otros medios orales de generación en generación por lo que la forma de aprender la lengua es variada. Esto provoca que haya diferencia en la forma de escribir entre las comunidades, incluso entre personas de una misma comunidad [Aikman, 1999]. Por esta razón, los textos que se escribieron en estas lenguas, como el shipibo-konibo, no dispusieron de un estándar ortográfico del cual guiarse, además que no tenían una necesidad de seguirlo.

Sin embargo, gracias al apoyo del gobierno para impulsar la inclusión social, se implementó el programa "Incluir para crecer" [Jara Males, Gonzales Acer, 2015] que establece que la enseñanza en los niveles de primaria y secundaria de zonas rurales debe ser enseñada en la lengua originaria del lugar además del español. Por lo que se genera una necesidad de recursos para la enseñanza ya que se presenta una deficiencia en la ortografía por la variedad de enseñanza de manera oral. Además se realizó una encuesta a nivel nacional [Ministerio de educación del Perú, 2013] que indica que en el país se ha incrementado el uso de las tecnologías en la educación. De manera que los alumnos podrían mejorar su rendimiento con ayuda de la tecnología, si es que esta contase con recursos computacionales adecuados, logrando así tener un impacto positivo.

Por lo descrito previamente, en este proyecto se afronta el problema de la carencia de apoyo y escases de recursos en la corrección ortográfica entre los hablantes de lenguas originarias en el Perú mediante la implementación un corrector ortográfico, utilizable desde una aplicación web. Para tener acceso al corrector y conseguir mayor difusión, se desarrollan servicios que

son consumidos en la aplicación web, en la cual se integra el corrector ortográfico y un módulo de sugerencias al usuario.

Dedicatoria

A mi familia que depositó su confianza y siempre estuvo para mí a lo largo de esta etapa.
A mis amigos y personas quienes me brindaron su apoyo a lo largo de esta etapa.
Finalmente a mi asesor Arturo, por su valioso apoyo y sus consejos durante el desarrollo de este proyecto que fue muy divertido e interesante.

Contenido

Índice de Figuras

Índice de Ecuaciones

Índice de Tablas

1. Introducción

1.1. Problemática

En nuestro planeta existen distintas lenguas, las cuales pueden ser agrupadas por múltiples criterios como su origen o su antigüedad. Sin embargo, muchas de estas lenguas no son tan difundidas o habladas por la población. Un ejemplo son las lenguas originarias de la Amazonía, las cuales sólo se hablan en ciertas zonas fronterizas entre Perú y Brasil. Muchas de las lenguas originarias en el mundo, son de tradición oral debido a que se desarrollan en pueblos de zonas rurales, donde la práctica oral funciona para mantener sus historias y tradiciones [Vansina, Udina, 2007].

En el Perú, existen diversas lenguas originarias como el asháninka, el kakataibo, entre otras [Rivera, 2001]. Estas lenguas se caracterizan porque son transmitidas a través de cuentos, poesía y otros medios orales de generación en generación, por lo que la forma de aprender la lengua es variada. Además, no cuentan con un proceso educativo que utilice textos del lenguaje, como por ejemplo un diccionario o un documento oficial. Esto provoca que haya diferencia en la forma de escribir entre las comunidades, incluso entre personas de una misma comunidad [Aikman, 1999]. Por esta razón, los textos que se escribieron en estas lenguas no dispusieron de un estándar ortográfico del cual guiarse, además que no tenían una necesidad de seguirlo.

En dicho contexto, la institución internacional ILV (Instituto Lingüístico de Verano[1]), enfocada en recopilar documentación sobre lenguas poco conocidas o minoritarias para su difusión, con ayuda del Ministerio de Educación, realizó una guía para el aprendizaje de las

[1] http://www.peru.sil.org/es

lenguas amazónicas en las cuales está incluida el shipibo-konibo [Faust, 1973]. Esta acción fue un primer acercamiento a la normalización del lenguaje.

Aún con estas iniciativas, se mantiene el problema de la deficiencia de la ortografía de los mismos hablantes, debido a que éstos reciben principalmente la enseñanza de su lengua de manera oral. Como es una lengua de tradición oral, los hablantes no le encuentran una funcionalidad a la escritura, por esa razón no producen textos [Vigil, 2005]. Sin embargo, el Gobierno Peruano está enfocándose en la educación en las lenguas originarias y la inclusión social con su programa *"Incluir para crecer"* [Jara Males, Gonzales Acer, 2015]. Un programa donde los colegios de educación primaria y secundaria enseñan las materias en la lengua originaria de la comunidad y en español. A su vez, se han desarrollado guías para el aprendizaje de la lengua, que serán utilizados como un estándar oficial para la enseñanza en los centros educativos.

Actualmente, en los colegios que dictan las clases en lenguas originarias, los alumnos tienen dificultades en el aprendizaje y genera problemas en la identificación de las palabras mal escritas [Vásquez, 2016]. Por lo cual, deben tener un reforzamiento en el tema de la ortografía, que es fundamental para la producción de material escrito [Rodriguez-Ortega, 2015]. Para mejorar la ortografía de los alumnos, los profesores de estos colegios sólo cuentan con libros de enseñanza, en los cuales los estudiantes pueden aprender mediante la resolución de ejercicios y la práctica constante. En este punto, los alumnos no cuentan con recursos suficiente, ni una persona que los pueda guiar en la resolución de sus tareas de ortografía, provocando que se mantenga un bajo rendimiento en esta materia [Rolando, 2012].

A pesar que el bajo rendimiento de los alumnos no tiene una relación directa con el uso de un corrector, hay países como Irlanda que utilizan los recursos computacionales para reforzar la enseñanza de los alumnos, por lo que se detecta que actualmente hay una tendencia a utilizar la tecnología y los nuevos recursos computacionales para la enseñanza, principalmente en nivel primaria [Ward, 2014]. Adicionalmente, en el Perú se realizó una encuesta a nivel nacional que indica que en el país se ha incrementado el uso de las tecnologías en la educación [Ministerio de educación del Perú, 2013]. En el caso de Irlanda, estos se enfocan en adaptar los recursos a un software de educación, de esa manera los estudiantes pueden realizar ejercicios de práctica y recibir la retroalimentación en un lenguaje más adecuado para su edad [Ward, 2014]. De esa manera los alumnos podrían mejorar con ayuda de la tecnología, si es que esta contase con recursos computacionales adecuados, logrando así tener un impacto positivo. Sin embargo, a diferencia del castellano que tiene gran cantidad de herramientas e información como lecturas de clase, aplicativos de educación, videos y ejercicios, el shipibo-konibo no cuenta con la misma magnitud de recursos evitando un incremento en la motivación del alumno por mejorar su ortografía.

Por lo descrito previamente, en este proyecto se plantea afrontar el problema de la carencia de apoyo y escasez de recursos en la corrección ortográfica entre los hablantes de lenguas originarias en el Perú mediante la implementación de una aplicación en conjunto con servicios web de corrección ortográfica. Este ofrecerá servicios a los usuarios que les permitirán configurar y utilizar el corrector ortográfico en las lenguas que deseen. Estos servicios tendrán funciones genéricas que permitirán adaptar los recursos a una lengua originaria. Además, para obtener el mayor alcance posible, la aplicación contará con una interfaz web, funciones de corrección, herramientas interactivas, en los cuales se verán reflejados el uso de los servicios, y contará como fuente la gramática desarrollada por el

gobierno, que por el momento es para el shipibo-konibo. Así mismo, debido a la carencia de expertos en las lenguas originarias, se contará con un componente de sugerencias en el cual el usuario podrá interactuar para que la corrección sea mejorada constantemente. Una vez implementado, sus servicios y funciones podrán ser utilizados para desarrollar futuras aplicaciones que lo requieran, como es el caso de aplicaciones educativas o de investigación.

1.2. Objetivos

1.2.1. Objetivo general

Implementación de una aplicación y servicio web de corrección ortográfica para lenguas originarias peruanas, específicamente para el shipibo-konibo.

1.2.2 Objetivos específicos

Para lograr el objetivo general se tienen los siguientes objetivos específicos:

- **Objetivo 1**: Implementar un algoritmo que efectúe la corrección de una palabra mal escrita en un lenguaje originario
- **Objetivo 2**: Implementar un componente de software que presente sugerencias de palabras correctamente escritas al usuario en un lenguaje originario
- **Objetivo 3**: Implementar un servicio y aplicación web que integre el algoritmo y componentes desarrollados para la presentación y validación de resultados

1.3. Resultados

Para cada uno de los objetivos específicos, se tienen los siguientes resultados esperados en la Tabla 1.

Tabla 1: Resultados

Objetivo	Resultado	Indicador
Implementar un algoritmo que efectúe la corrección de una palabra mal escrita en un lenguaje originario.	Algoritmo para la corrección de palabras desarrollado.	Pseudocódigo del algoritmo.
	Reporte de experimentación con métricas de evaluación para correctores ortográficos.	Reporte de experimentación con métricas de evaluación para correctores ortográficos.
	Lista de palabras mal escritas y corregidas.	Reporte de experimentación con las palabras utilizadas.
Implementar un componente de software que presente sugerencias de palabras correctamente escritas al usuario en un lenguaje originario.	El componente de software para identificar las sugerencias que se mostrarán al usuario a través de un ranking de aparición.	Pseudocódigo del componente. Manual de uso.
	Lista de palabras mal escritas y sugeridas.	Reporte de experimentación con las palabras sugeridas.
Implementar un servicio y aplicación web que integre el algoritmo y componentes desarrollados para la presentación y validación de resultados.	Servicio web que integra las los algoritmos y el componente de software.	Manual de uso
	Aplicación web que integra las los algoritmos y el componente de software.	Manual de uso
	Reporte de evaluación y funcionamiento del corrector ortográfico.	Reporte de experimentación

1.4. Herramientas, métodos y procedimientos

Se detallarán las herramientas, métodos, metodologías y procedimientos que se utilizarán para conseguir los resultados esperados. Todo lo antes mencionado se puede ver reflejado en la Tabla 2 donde se agrupa por cada resultado esperado.

Tabla 2: Herramientas, métodos y procedimientos a utilizar

Resultados esperados	Herramientas, métodos y procedimientos
Algoritmo para la corrección de palabras desarrollado.	● Python ● Neo4j ● NLTK ● MySQL
Reporte de evaluación del funcionamiento del algoritmo.	● *Edit distance* ● *Precision* ● *Recall*
Reporte con la lista de palabras corregidas.	● Python
El componente de software para manejar las sugerencias que se mostrarán al usuario.	● Python ● Django
Reporte con la lista de palabras sugeridas.	● Python
Servicio web que integra los algoritmos y el componente de software.	● Python ● Django ● Scrum
Aplicación web que integra el algoritmo y el componente de software.	● Python ● Django ● Scrum
Reporte de funcionamiento del corrector ortográfico.	● *Edit distance* ● *Precision* ● *Recall*

1.4.1. Herramientas

1.4.1.1. Python

Python es un lenguaje de programación que combina una sintaxis sencilla con potencia. Esta contiene módulos, clases, excepciones, tipo de datos de alto nivel y tipado dinámico. Además de interfaces a librerías y llamadas al sistema en las distintas implementaciones de este. [Python.org, 2016]

Se eligió este lenguaje debido a su recurrido uso para el procesamiento de lenguaje natural y por ser multiplataforma. Además, porque este posee múltiples librerías que serán de ayuda al desarrollo de los objetivo. Otra razón fue que existe amplia documentación respecto al lenguaje que facilitará el dominio del lenguaje para realizar el proyecto.

1.4.1.2. Django

Django es un framework web de Python, este fomenta el desarrollo rápido y el diseño limpio. Ayuda a centrarse en el desarrollo de la aplicación sin la gran molestia del desarrollo web. [Django Project, 2016]

Se eligió este framework ya que ofrece una manera fácil de trabajar con bases de datos relacionales y porque tiene una estructura sencilla y rápida de entender. Además, éste ayudará al proyecto porque ya ofrece una estructura para el desarrollo de aplicaciones web y un buen manejo en el diseño de las vistas de manera que se mostrará interfaz que se utilizará para probar el corrector.

1.4.1.3. Natural Languague Toolkit (NLTK)

NLTK es una librería que facilita la creación de programas en Python para trabajar con lenguaje natural. Ofrece un conjunto amplío de recursos léxicos y bibliotecas de procesamiento [Natural Languague Toolkit, 2016].

Se eligió ésta librerías por que ofrece una gran cantidad de funcionalidades que ayudarán al tratamiento del lenguaje natural y no se tendrá que crear funciones que ya existentes.

1.4.1.4. Regex (re)

Regex es un módulo interno del lenguaje Python que ayuda a trabajar con patrones de expresiones regulares. Estos ayudan a crear expresiones regulares en el tratamiento de textos.

Se aprovechará éste módulo porque será de ayuda para el tratamiento y normalización del texto a trabajar. De esta manera se podrá manejar una previa limpieza de los datos, para su posterior procesamiento y corrección.

1.4.1.5. MySQL

MySQL Community Edition es una base de datos open source, cuya versión es muy popular en el mundo. Además está disponible bajo la licencia GPL y es soportada por muchas comunidades de desarrolladores open source. [MySQL, 2017]

Se eligió esta base de datos por su facilidad de uso y la basta documentación de la que dispone. Otro punto que se consideró es que trabaja directamente con el framework elegido previamente.

1.4.1.6. Neo4j

Neo4j es una base de datos de grafos altamente escalable que aprovecha las relaciones de datos como entidades, para crear aplicaciones inteligentes y satisfacer los actuales desafíos de datos en constante evolución. [Neo4j, 2017]

Se eligió esta base de datos de grafos, por su facilidad de uso y porque será necesario para almacenar ciertos grafos que se requerirán para el uso en el algoritmo. Otro punto que se consideró es que trabaja con el framework elegido previamente.

1.4.2. Métodos y procedimientos

1.4.2.1. Métrica de *Edit Distance*

La "*Edit distance*" o distancia de edición es un método para obtener el menor número de ediciones (sustituir, eliminar, insertar) que debes realizar a una cadena de caracteres para que sea transformado en otra. [Przybocki, Sanders, Le, 2006]

Se eligió este método de evaluación de los reportes porque es usado concurrentemente como evaluación de algoritmos que tratan con la traducción de palabras y corrección de palabras. Además, es un buen indicador para identificar la cantidad de operaciones que realiza el algoritmo.

1.4.2.2. Métrica de Precisión (*Precision)*

La "*precision*" o precisión es una medida que ayuda a evaluar los resultados identificando la relevancia de los resultados. Un ejemplo aplicado al proyecto sería que de una oración de 10 palabras, de las cuales 4 están mal escritas. El algoritmo las identifique y luego sean corregidas correctamente, en ese caso se obtendría el más alto valor de precisión [Raghavan, Bollmann, Jung, 1989].

Se eligió éste método de evaluación para identificar el rendimiento del algoritmo de corrección y mostrar la eficacia de su trabajo ya que esta métrica es comúnmente utilizada para evaluar los correctores ortográficos [van Huyssteen, et al., 2004].

1.4.2.3. Métrica de Exhaustividad (*Recall)*

El "*recall*" o exhaustividad es una medida que ayuda a evaluar los resultados identificando la proporción de casos positivos que el algoritmo ha predicho como positivos [Powers, 2011].

Se escogió éste método de evaluación para identificar el rendimiento del algoritmo de corrección y mostrar la eficacia de su trabajo debido a que estas métricas suelen utilizarse para evaluar los correctores ortográficos [van Huyssteen, et al., 2004].

1.4.3. Metodologías

1.4.3.1. *Scrum*

Scrum es una metodología ágil que aporta un conjunto de buenas prácticas y roles para el desarrollo de software, reduciendo considerablemente la documentación que se suele utilizar en otras metodologías. Esta está enfocada más en el trabajo de desarrollo y la entrega de valor al producto.

Dentro de todas las prácticas que ofrece esta metodología se ha elegido utilizar:

- Product backlog: Documento con la lista de requisitos para la aplicación.
- Trabajo con iteraciones: A través de las iteraciones se irán realizando los avances de la aplicación.

- Sprint review meeting: Reunión al finalizar una iteración para evaluar los avances y problemas que han surgido durante el desarrollo. De manera que se pueda tomar medidas necesarias para alcanzar los objetivos esperados.

Se ha escogido esta metodología ágil para elaboración de la aplicación y servicios web porque ésta se enfoca más en el desarrollo y programación, lo cual beneficiará para acelerar el desarrollo del algoritmo de corrección de palabras mal escritas. A su vez, esta metodología cuenta con bastante documentación de la cual apoyarse [Schwaber, Beedle, 2002].

1.5. Alcance, limitaciones y riesgos

1.5.1. Alcance

Este proyecto de fin de carrera pertenece al área de ciencias de la computación, en la rama del procesamiento de lenguaje natural. En él, se propone desarrollar un servicio y aplicación web que permita afrontar el problema de carencia de apoyo y escases de recursos en la corrección ortográfica entre los hablantes de lenguas originarias en el Perú, principalmente enfocado en el shipibo-konibo. Este consta en esencia de un algoritmo para la detección de una palabra mal escrita y su corrección, además de un componente de software que permite ofrecer sugerencias de palabras al usuario que va mejorando las sugerencias conforme se utiliza.

Este proyecto sólo se enfocará en la lengua shipibo-konibo. Asimismo, al realizar la corrección ortográfica, no se espera tener una corrección ortográfica altamente precisa, debido a la dificultad y los tipos de errores ortográficos que se manejarán que son el error de agregar una letra adicional, eliminar una letra, cambiar una letra y permutación de letras adyacentes.

Para el proyecto se hará una página web dónde se integrarán el algoritmo de corrección desarrollado y el componente de sugerencias de palabras correctas, esto es así porque de esta manera se podrá interactuar con el usuario para mejorar el corrector.

1.5.2. Limitaciones

Para la realización de este proyecto se han identificado una limitación referente al tamaño del corpus ya que este es pequeño y no cuenta con muchos datos. Por lo que se ha planeado ir alimentándolo con ayuda del proyecto. Otra limitación que se considera es la falta de conocimiento del shipibo-konibo por lo que se ha optado por contar con ayuda adicional de hablantes de la lengua.

1.5.3. Riesgos

En la Tabla 3 se muestran los riesgos que se pueden presentar durante el desarrollo del proyecto.

Tabla 3: Riesgos del proyecto

Riesgo	Principal objetivo afectado	Impacto	Estrategia de mitigación
El corpus recolectado en el proyecto no es suficiente.	Implementar un algoritmo para la identificación de una palabra mal escrita en un lenguaje originario según el tipo de error.	Medio	El autor de la tesis ayudará en la creación del corpus adicional.
Falta de disponibilidad de tiempo de los expertos.	Implementar un algoritmo para la identificación de una palabra mal escrita en un lenguaje originario según el tipo de error	Alto	Uso de distintos medios digitales para recibir ayuda de los expertos.

1.6. Justificación

Según lo expuesto en el presente proyecto de tesis, la deficiencia de la ortografía entre los hablantes de lenguas originarias del Perú es el principal problema que se desea afrontar.

Este proyecto será beneficioso para alumnos hablantes del shipibo-konibo, ya que contarán con un aplicativo web que utiliza la gramática oficial desarrollada por el gobierno como base,

y donde podrán desarrollar prácticas que le servirán como apoyo adicional en su proceso de aprendizaje.

A su vez, ayudará a la comunidad académica que pretenda utilizar el corrector ortográfico como una herramienta adicional para un proyecto de investigación más grande, ya que este proyecto contará con servicios web que se pondrá a disposición de quien lo requiera.

Otro aspecto beneficioso es que el proyecto será un recurso computacional adicional para la lengua shipibo-konibo, la cual, como se menciona, es una lengua de escasos recursos computacionales, de manera que motivará a que se sigan desarrollando recursos para esta lengua y las distintas lenguas originarias del Perú.

2. Estado del arte

Revisar la literatura relacionada es un paso necesario para realizar la implementación de una aplicación y servicios web de corrección ortográfica independiente del lenguaje debido a que se necesita obtener el conocimiento necesario acerca de propuestas similares. Por ello, el objetivo principal de la revisión del estado del arte es estudiar y comprender los enfoques que se han utilizado para tratar la implementación de un una aplicación y servicios web de corrección ortográfica en el tiempo y así tener una base para el proyecto.

Para la realización del estado del arte se aplicó una revisión sistemática. Se escogió este método a diferencia del método tradicional porque la revisión sistemática ayuda a minimizar la imprecisión en las búsquedas mediante una metodología científica.

Primero, se plantearon las siguientes preguntas de investigación: ¿Qué métodos independientes del lenguaje se utilizan para corregir la mala ortografía en textos? y ¿Qué métodos se aplican para la corrección ortográfica de lenguas de escasos recursos?

En base a las preguntas planteadas se determina ciertas palabras clave que ayudan a resolver las preguntas de investigación. La lista de términos es:

- Spell-checker
- Spelling correction
- Agglutinative language (característica del lenguaje shipibo-konibo)
- Independent language
- Low-resource languages
- Minority languagues
- Under-resource languages

Una vez definida la lista de términos se generaron las siguientes cadenas de búsqueda:

Spell-checker AND “Independent language”

Spell-checker AND “Low-resource languages”

Spell-checker AND “Minority languages” AND “Agglutinative”

Estas cadenas de búsqueda se utilizaron en librerías digitales como ACM[2], Scopus[3], IEEE Xplore[4], Web of Science[5] y ScienceDirect[6]. Por último, con los resultados generados se revisaron, en primera instancia, los títulos y los resúmenes de los artículos para tener una idea general del tema que se trata en los estudios, y filtrar aquellos que no son relevantes. Posteriormente, se procede a revisar el documento completo con el fin de hallar algún conocimiento que aporte a la solución del problema.

2.1. Trabajos relacionados

Luego de una revisión de las publicaciones obtenidas, se identificaron 4 fuentes principales que ayudan a responder a las preguntas planteadas.

En primer lugar, para la pregunta formulada: ¿Qué métodos independientes del lenguaje se utilizan para corregir la mala ortografía en textos?, la primera fuente relevante es la de Barari y QasemiZadeh (2005), quienes presentaron una herramienta denominada “CloniZER Spell Checker Adaptative”. Ésta consiste en un método adaptativo que utiliza reconocimiento de

[2] https://www.acm.org/
[3] https://www.scopus.com/
[4] http://ieeexplore.ieee.org/
[5] https://apps.webofknowledge.com/
[6] http://www.sciencedirect.com/

patrones de error interno que está basado en la estructura conocida como árbol ternario de búsqueda. Gracias a ese enfoque, su corrector era independiente del lenguaje ya que no se basaba en reglas específicas de una lengua o en un algún corpus específico, siendo este reemplazable. Un enfoque interesante es que con ayuda del árbol, que posee aristas de peso variable, se va realizando modificaciones a través de la interacción con el usuario, ya que su método aprende de una media de patrones de error y así la sugerencia de soluciones va mejorando. En la Figura 1 se ve un ejemplo más detallado de la secuencia y trabajo que realiza su corrector.

Por otro lado, otro método que se identificó fue el de Abdullah y Rahman (2003), quienes realizaron un motor de corrección ortográfica genérico para lenguas del sur de Asia. Este utiliza listas circulares donde las palabras se agrupan por similitud fonética, de manera que con un algoritmo adaptado del Recursive Simulation [Lee, 1997] se va construyendo la palabra posiblemente correcta que tiene parecido a la palabra mal escrita. Sin embargo, una ayuda interesante que utilizaban era un diccionario adicional de palabras, en donde guardaban las palabras con errores que escribían los usuarios. Este método es favorable para las lenguas que tengan similitud con otras fonéticamente, como por ejemplo el grupo de lenguas de la familia Pano en la que se encuentra el shipibo-konibo.

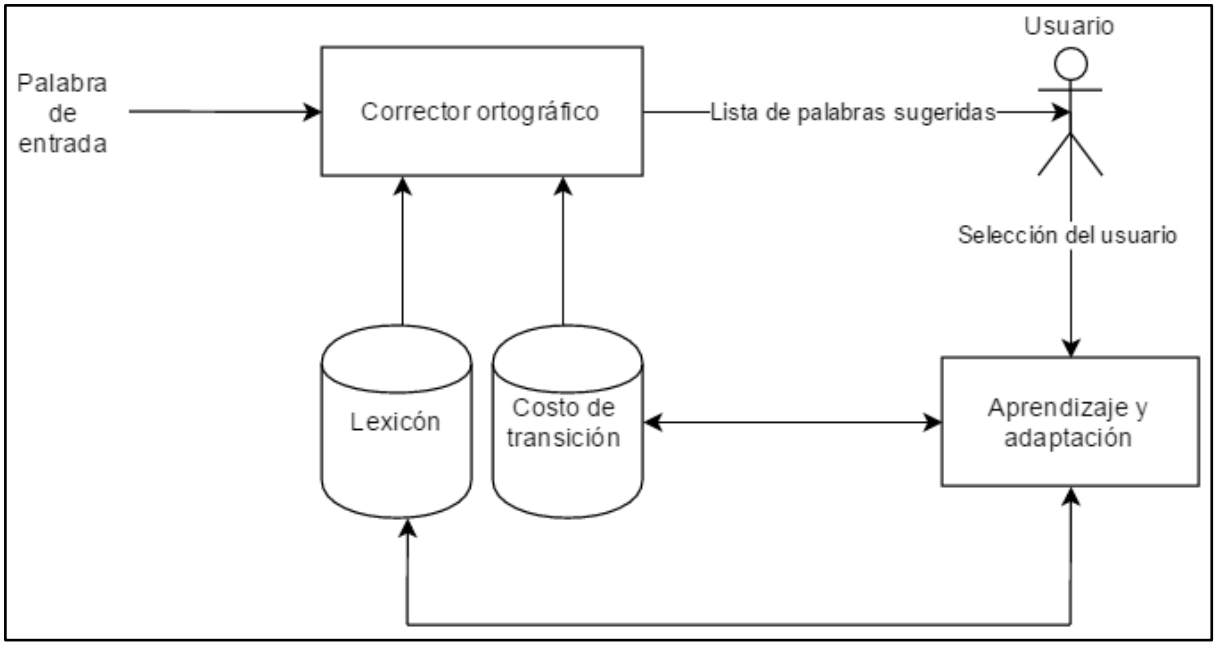

Figura 1: Esquema de módulos del corrector ortográfico CloniZER. Imagen adaptada de [Barari, QasemiZadeh, 2005]

En segundo lugar, respecto a la pregunta: ¿Qué métodos se aplican para la corrección ortográfica de lenguas de escasos recursos?, una de las fuentes que se encontró como respuesta es la de Aduriz, Urkia, Alegria, Artola, Ezeiza y Sarasola (1997), quienes presentaron un corrector basado en la morfología. Utilizan un análizador morfológico, el cual consiste en realizar descomposiciones morfológicas en dos niveles para errores ortográficos, y para los errores tipográficos utiliza un reconocimiento de morfemas en la generación de palabras correctas. Este enfoque es interesante ya que adicionalmente utilizan un lexicón que van mejorando y un conjunto de reglas que ayudan a mapear el nivel léxico y el nivel de superficie debido a la transformación morfo-fonológica (representación fonológica de los morfemas). Posteriormente, el corrector de la fuente derivó en un software comercial llamado Xuxen [Elhuyar Hizkuntza eta Teknologia, 2016].

Otra fuente que contribuyó a responder la pregunta es de Wasala, Weerasinghe, Pushpananda, Liyanage, y Jayalatharachchi (2011), que presentaron un corrector para una lengua africana. En este se utilizó un modelo estadístico basado en *n-grams*. Este enfoque se basa en asignar probabilidades a una secuencia de palabras donde la secuencia se determina

por el *n-gram*. Un ejemplo de un *2-gram* o *bigram es: "prueba esto"*. El enfoque elegido ofrece relativa facilidad de construcción además evita el problema de contar con escasos recursos lingüísticos.

El algoritmo que se propone para la corrección ortográfica utiliza 4 módulos como se observa en la Figura 2. Estos son: pre-procesamiento, generación de permutaciones, selección de mejores sugerencias, y post-procesamiento. Lo interesante de este algoritmo es que luego del pre-procesamiento realiza una búsqueda de palabras con similar sonidos o fonemas, generando así posibles soluciones, que luego son mejoradas en base a las estadísticas de los *n-grams* aplicando desde unigramas hasta trigramas.

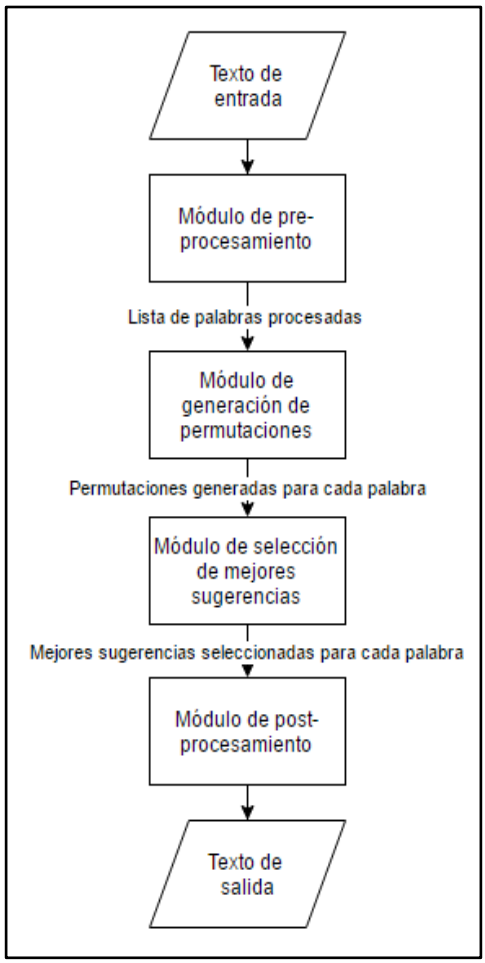

Figura 2: Algoritmo propuesto para la corrección ortográfica del Sinhala. Imagen adaptada de [Wasala, Weerasinghe, Pushpananda, Liyanage, Jayalatharachchi, 2011]

2.2. Observaciones y conclusiones

Luego de haber revisado los distintos trabajos, se pudo responder a las preguntas planteadas con la que se descubrió que existen distintos tipos de enfoque para afrontar el problema de la corrección ortográfica independiente del lenguaje. Desde el enfoque estadístico basado en *n-grams,* que es utilizado para lenguas de escasos recursos, hasta la utilización de relaciones morfológicas y corpus etiquetados que se utilizan para correcciones independientes del lenguaje, además de los distintos tipos de errores ortográficas que se pueden tener como el error de competencias que es inherente al usuario al no saber escribir y el error de tipografía que el usuario puede cometer al mecanografiar el texto.

Dentro de los trabajos esencialmente se necesita contar con un lexicón o diccionario de las palabras escritas, un modelo para el trato de las palabras y la aplicación de algoritmos para el tratamiento del modelo.

Esto será de mucha ayuda para el proyecto para tener una base de conocimiento previo y plantear un enfoque que puede combinar más de uno de los identificados, como el tratamiento de las palabras por los fonemas y el estadístico basado en *n-grams, a*demás de determinar las funciones que se van a desarrollar para recolectar el texto, su tratamiento y la muestra de generar los resultados.

Con respecto a los trabajos identificados en el estado del arte, este proyecto proveerá de una herramienta web libre para la utilización y adaptación a otras lenguas originarias peruanas para la corrección ortográfica. Incluyendo la corrección ortográfica a través de 2 opciones. El primero, que será la corrección ortográfica ingresando un texto y obteniendo el resultado, o la segunda opción, a partir de una interacción con el usuario conforme se vaya escribiendo. De esta manera se lograría mejorar la predicción de la palabra que el usuario desea corregir.

3. Marco teórico

En este capítulo se presentarán los conceptos necesarios para lograr un entendimiento del problema a resolver.

Conceptos relacionados a Lingüística

En esta sección se presentarán conceptos relacionados a las lenguas y su estudio:

1. Lingüística

   Estudio teórico del lenguaje que se ocupa de métodos de investigación y de cuestiones comunes a las diversas lenguas [RAE, 2014].

2. Lengua

   Es un sistema de comunicación verbal propio de una comunidad. [RAE, 2014]. En este estudio se hará referencia a tipos de lenguas como aglutinantes, de escasos recursos y minoritarias.

3. Lenguas aglutinantes

   Es una lengua en la cual sus palabras son formadas por una raíz y distintos afijos [Diaconescu, et al., 2009]. Para mencionar algunos ejemplos de lenguas aglutinantes está el shibipo-konibo, aimara o el vasco.

4. Lenguas de escasos recursos

   Son lenguas que poseen recursos computacionales limitados y con escasos recursos de entrenamiento. [Hartmann, et al., 2014]

5. Lenguas minoritarias

   Son lenguas habladas por grupos minoritarios que tienen un carácter histórico o social [Allardt, 1984].

6. Lenguaje

Es un método de comunicación por medio de un sistema de símbolos producidos de manera deliberada. Estos símbolos son ante todo auditivos [Sapir, 1966].

7. Ortografía

   Conjunto de normas que regulan la escritura de una lengua [RAE, 2014].

8. Morfología

   Es una rama de la gramática que se encarga del estudio de la formación de palabras e inflexión a partir de los morfemas [Aronoff, 1994].

9. Morfemas

   Son las mínimas unidades lingüísticas con un significado léxico o gramatical [Booij, 2012]. Un ejemplo se da en la construcción de la palabra gato con el lexema gat- y en conjunto con sus morfemas (-o, -a, -o -s, -a -s) se genera gato, gata, gatos, gatas.

Tipos de errores de ortografía

En esta sección se definirán los distintos errores que un corrector ortográfico podría detectar [Tolentino, et al., 2007]:

1. Error tipográfico

   Este error abarca 4 tipos de errores el de adición, eliminación, cambio de letra e intercambio de adyacentes.

2. Error cognitivo

   Error producido por el desconocimiento de la palabra a teclear.

Un ejemplo del primer tipo de error se da cuando se desea teclear “casa”, pero por error al teclear se presiona “d” en lugar de “s” y resulta “cada”. En cambio un ejemplo del segundo

tipo de error se da cuando se desea teclear “ahora”, pero por desconocimiento se teclea “aora”.

Distancias entre textos

En esta sección se explicarán las distancias que se utilizan para comparar las palabras [Deza, Deza, 2009]:

1. Distancia Euclidiana

   Es la distancia geométrica aplicada de un espacio multidimensional en la cual se comparan los caracteres de dos palabras. Definido como:

$$||x - y||^2 = (x_1 - y_1)^2 + \cdots (x_n - y_n)^2$$

Ecuación 1: Distancia euclidiana

2. Distancia Levenshtein

   Es una distancia que se obtiene por el número mínimo de operaciones con caracteres para que una palabra sea igual a otra, donde cada operación se puede trasponer, sustituir, insertar o eliminar caracteres. Un ejemplo es la distancia entre: “cero” y “caor”, donde se considera que la distancia es 3. 1 por cambiar “e” por “a” y 2 por la transposición de “or” en “ro”.

3. Distancia Damerau-Levenshtein

   Es una variante de la distancia Levenshtein que considera la transposición de caracteres como una sola operación. El mismo ejemplo para hallar la distancia entre: “cero” y “caor”, ahora se considera que la distancia es 2. 1 por cambiar “e” por “a” y 1 por la transposición de “or” en “ro”.

Procesamiento del lenguaje natural

En esta sección se presentarán conceptos relacionados al procesamiento del lenguaje natural:

1. Procesamiento del lenguaje natural (PLN):

   Este consiste en utilizar el lenguaje natural para realizar una comunicación con la computadora, logrando que la computadora lo reconozca para realizar alguna tarea relacionada con el lenguaje [Vásquez, et al., 2009].

2. Recursos computacionales

   Son los recursos utilizados por modelos computacionales para resolver distintos problemas [Sow, Eleftheriadis, 1998].

3. Lexicón

   Un lexicón especifica las propiedades de cada palabra, su forma fonológica, sus propiedades morfológicas y sintácticas, y su significado [Booij, 2012].

4. Corpus anotados

   Es una colección de datos sobre el uso concreto la lengua [Booij, 2012].

5. Corrector ortográfico

   Consiste en verificar que cada palabra de un texto pertenece al lenguaje en el que está escrito y de ser necesario, proveer de posibles correcciones [Ndiaye, Faltin, 2003].

6. Normalización del lenguaje

   Consiste en convertir el texto a una forma estandarizada más conveniente para su uso [Jurafsky, Martin, 2014].

Otros conceptos del contexto del problema

1. Grupo étnico

Es una comunidad que comparte valores culturales y que cuenta con miembros que se identifican a sí mismos, así como son identificados por otros y que constituyen una categoría distinguible de otras categorías del mismo orden [Barth, 1976].

2. Shipibo-konibo

   Es un grupo étnico que forma parte de la familia lingüística Pano, ubicados a en las fronteras de Perú, Bolivia y Brasil [Morín, 1998].

3. Lenguas originarias

   Son lenguas que han existido antes a la difusión del español, que han mantenido y aún son utilizadas en el ámbito del territorio nacional [Acosta, et al., 2013].

4. Diseño de estructuras de datos para el algoritmo de corrección ortográfica

En este capítulo se abordará las estructuras que se utilizarán a lo largo del desarrollo del proyecto. Además de ciertas funciones adicionales que son necesarias para el funcionamiento del algoritmo de corrección.

4.1. Estructuras

Las estructuras fueron enseñadas en los cursos como grafos y listas, las cuales utilizan distintos unidades como caracteres, sílabas y palabras.

4.1.1 Grafo de letras

La estructura que se necesita inicialmente consta de un grafo que será almacenado en una base de datos especial para grafo que es Neo4j:

Este grafo tiene como nodo las letras que se utilizan en el shipibo-konibo y los vértices la relación de 2 nodos. Las letras del shipibo-konibo constan de 4 vocales: a, e, i, o y de 15 consonantes: b, k, ch, j, m, n, p, r, s, sh, x, t, ts, w, y. Sin embargo, como se plantea utilizar un enfoque basado en *n-grams*, los nodos que se utilizarán varían en la cantidad de caracteres.

- Nodo:
  - Caracter
  - Número de caracteres
  - Veces que aparecen en el diccionario
- Vértice:
  - Valor de la unión
  - Posición

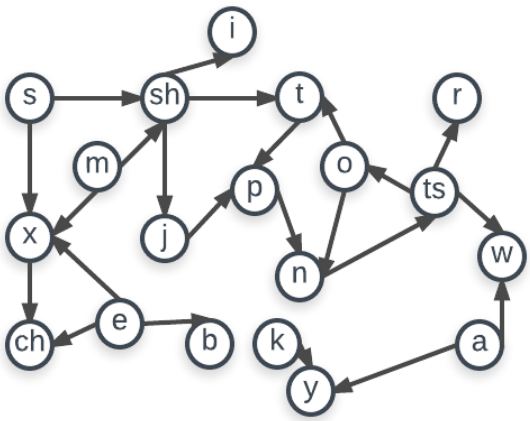

Figura 3: Representación de grafo de 1 carácter

4.1.2 Grafo de sílabas

La estructura que se necesita para mejorar las posibles soluciones del algoritmo consta de un grafo que será almacenado en una base de datos especial para grafo que es Neo4j:

Este grafo tiene como nodo las sílabas que existen en el shipibo-konibo y los vértices la relación de 2 nodos. El número de sílabas del shipibo-konibo que hay son 576, con estas sílabas se pueden generar todas las palabras del diccionario.

- Nodo:
  - Sílaba
  - Veces que aparecen en el diccionario
- Vértice:
  - Valor de la unión
  - Posición

4.1.3 Lista de palabras corregidas

Se utilizará una lista como un diccionario que asocia una palabra mal escrita que ya ha sido corregida previamente para evitar corregir nuevamente un error y evitar realizar un trabajo

adicional. Para esto se tiene como llave la palabra mal escrita y esta es asociada a una lista de palabra corregidas por los usuarios previamente.

Llave:
Palabra mal escrita
Lista:
Palabra bien escrita
Cantidad de veces corregida

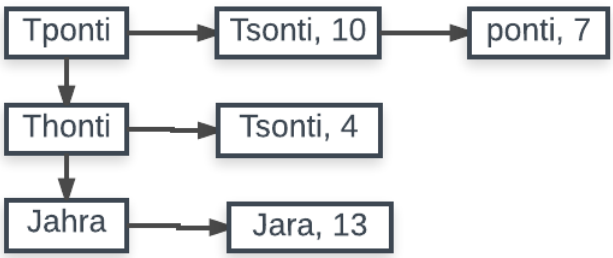

Figura 4: Estructura de términos corregidos

4.2. Funciones

4.2.1 Silabificador

Una función que ayuda en la mejora de las soluciones que halla el algoritmo de corrección es el silabificador, este ha sido desarrollado en base a las reglas que existen para la creación de palabras en el shipibo-konibo. El uso del silabificador permitió separar cada palabra del diccionario en sílabas para crear la estructura, además que es un filtro que se utiliza para identificar si una palabra tiene una escritura consistente de acuerdo al lenguaje.

5. Implementación del algoritmo de corrección ortográfica

En este capítulo se abordará el desarrollo del algoritmo de corrección de palabras mal escritas en el lenguaje originario seleccionado: shipibo-konibo. Este algoritmo consta de 2 partes, la primera consiste en un pre-procesamiento e identificación de palabras mal escritas, mientras que la segunda es la corrección de las palabras identificadas como mal escritas.

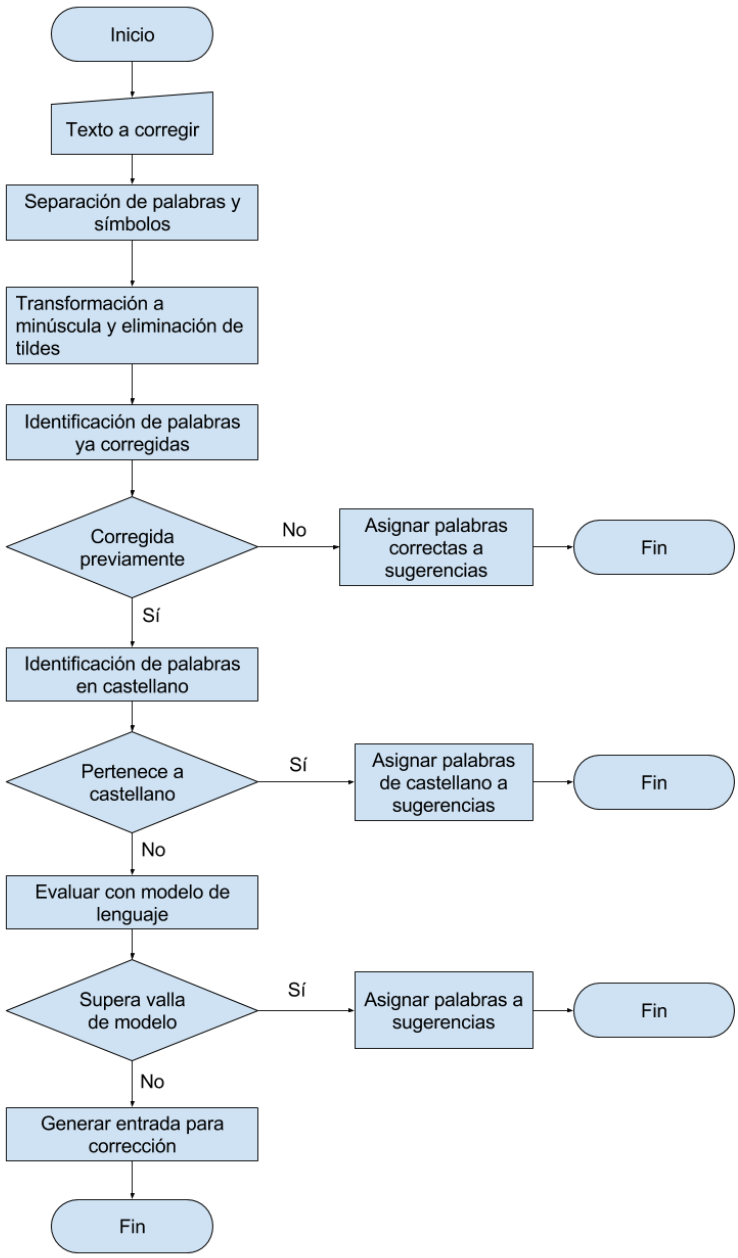

Figura 5: Diagrama de identificación de palabras mal escritas

5.1. Identificación de palabras mal escritas

Como se ve en la Figura 5, primero se realiza un pre-procesamiento de todo el texto que se planea corregir, separando las palabras y los elementos que no son palabras como los números o los signos de puntuación. Ya separadas las palabras y caracteres especiales se procede a guardar su posición para su futura colocación en el texto corregido.

Luego, las palabras se proceden a transformar a minúscula, ya que el grafo que se utiliza para el siguiente algoritmo no es sensible a mayúsculas y minúsculas. Una vez realizada la transformación a minúscula, se procede a retirar las tildes ya que en el siguiente algoritmo se evaluará si una palabra requiere tilde.

Como en la siguiente parte del algoritmo se realiza la corrección, se decidió guardar en una estructura las palabras ya corregidas para evitar un procesamiento adicional en la corrección. En esta estructura se realiza una búsqueda directa de la palabra mal escrita y se obtienen las palabras correctas relacionadas, para que finalmente el usuario pueda realizar su elección.

Las palabras que no son encontradas en esta estructura, son posteriormente evaluadas para identificar si dentro de las palabras, se encuentra alguna tomada del castellano. Esto se realiza debido a que el alcance de la tesis es corregir palabras sólo de lenguas amazónicas.

Para evaluar si una palabra está en castellano, se utilizó una búsqueda en un corpus de palabras en castellano, en el que se seleccionan las palabras que se encontraban en el corpus y son asignadas a la lista de sugerencia. Las palabras que no son encontradas en este corpus, se

proceden a evaluar mediante un modelo de lenguaje de sílabas, las cuales pasan a ser la entrada para la corrección en caso no superen la valla.

5.2. Corrección de palabras mal escritas

Con la entrada de palabras mal escritas se inicia la corrección. Para esto, se carga un grafo cuyos nodos corresponden a las letras que se utilizan en el alfabeto oficial del shipibo-konibo, mientras que los vértices representan la relación que se da entre estas letras. Esta relación es el número de veces que 2 letras están juntas en una palabra del diccionario. El grafo considera la repetición de cada letra y la repetición de las conexiones entre letras que se dan al procesar el diccionario de palabras en shipibo-konibo. Luego, se carga un segundo grafo que tiene como nodos todas las sílabas posibles del shipibo-konibo y los vértices, la relación que se da entre estas sílabas. De manera similar al primer grafo, el segundo considera la repetición de cada sílaba y la repetición de las conexiones entre las sílabas que se dan en las palabras al procesar el diccionario.

Una vez que se cargan los grafos, se evalúa la palabra a corregir y de acuerdo a la cantidad de vocales se puede identificar una cantidad aproximada de sílabas que se tendrá. Como la ausencia o aumento de una vocal puede ser un error, se considera un intervalo de sílabas que puede tener una palabra [# de sílabas -1; # de sílabas + 1]. Esto ayuda a la hora de generar las posibles soluciones de una palabra.

Luego de calcular el intervalo, se procede a realizar la búsqueda de las soluciones. Esto se realiza recorriendo de manera recursiva la palabra mal escrita. Conforme se va avanzando letra por letra se va formando una solución verificando que la solución formada cumpla una silabificación y el intervalo de sílabas y calculando el valor de esa solución formada mediante

la suma de la repetición de cada letra en el diccionario y el valor de su unión en el grafo. Mientras se recorre la palabra mal escrita al encontrar un error ortográfico, se generan caminos distintos, el primero a partir de eliminar la letra errónea, el siguiente es al realizar un cambio de letra con la anterior, y los demás caminos se generan al cambiar la letra errónea con las letras que en el grafo están relacionadas a letra adyacente la errónea. De esta manera, al crearse varios caminos se van generando distintas soluciones.

Una vez finalizada la búsqueda de soluciones correctas, estas se evalúan a través de un modelo de lenguaje modificado. Para eso se utiliza el segundo grafo que contiene sílabas y las repeticiones de las conexiones entre sílabas. Con el objetivo de iniciar el proceso se separa cada posible solución en sílabas y con ayuda del grafo se calcula la suma de las conexiones de estas sílabas. Adicionalmente, se calcula la distancia de edición con Damerau-Levenshtein. Al terminar de calcular los dos valores se realiza una multiplicación de 90% al Damerau-Levenshtein y 10% a la suma de relaciones de sílabas, estos valores de 90% y 10% se escogieron luego de realizar pruebas para identificar cuál optimizaba los resultados, finalizados los cálculos a cada palabra se escogen las tres posibles palabras correctas con los mejores valores para ser retornadas como solución.

6. Implementación del componente de sugerencias al usuario

En este capítulo se abordará el desarrollo del componente de sugerencias al usuario. Este tiene la función de ofrecer sugerencias de posibles palabras correctas y al ser alguna la seleccionada, estos cambios ayudarán a mejorar las estructuras internas que se utiliza en el algoritmo para mejorar el corpus y la corrección misma en ejecuciones posteriores.

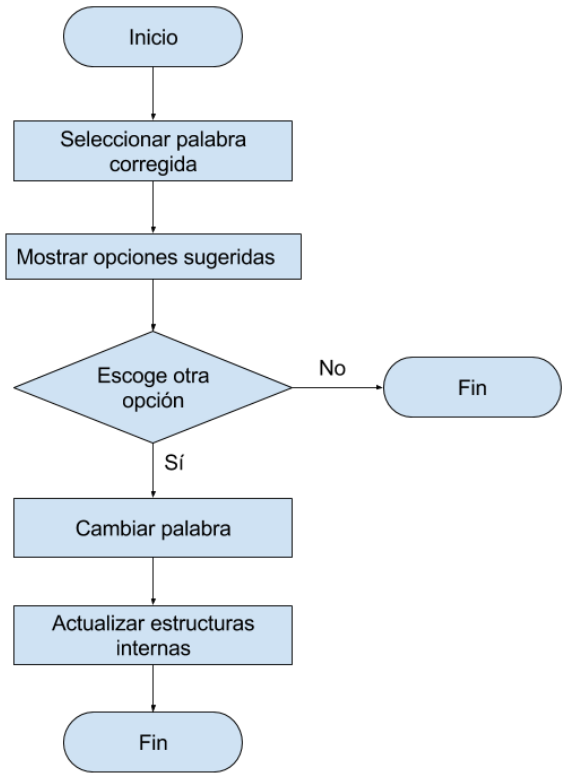

Figura 6: Diagrama de componente de sugerencias

El componente de sugerencias se ha definido para que el usuario pueda realizar correcciones a la solución que el aplicativo le brinda, permitiendo mejorar los resultados ya que no se tendrá una corrección totalmente precisa y adicionalmente agregando elementos nuevos a la lista de corregidos. Este componente será desarrollado con JavaScript, para la parte en la que

interactúa con el usuario, y en complemento con Python en la parte de mejora de las estructuras.

Como se ve en la Figura 6, se inicia cuando el usuario selecciona la palabra corregida, activando un menú con las palabras sugeridas adicionales que se obtuvieron al realizar la corrección. Luego, el usuario selecciona la opción que le parece más acertada y esta se cambia en el texto corregido. Una vez realizado este cambio, se realiza un cambio adicional en las estructuras internas, donde se agrega la palabra correcta a la lista de corregidos y se actualizan los valores en el grafo interno que se maneja para el algoritmo de corrección.

7. Integración de algoritmo de corrección y componente de sugerencias

En este capítulo se detallará la integración del algoritmo de corrección y el componente de sugerencias para finalizar el proyecto, adicionalmente se mostrará el manual de uso del aplicativo y servicio web que será el resultado final del proyecto.

7.1. Base de datos

7.1.1 MySQL

Para almacenar un corpus de palabras en castellano, se utiliza un modelo de base de datos simple como se observa en la Figura 7, en el cuál se realizará una búsqueda simple para hallar las palabras del castellano que se encuentre en el texto a corregir.

Palabra:
- ID
- Término (en castellano)
- Veces

Figura 7: Modelo de corpus español

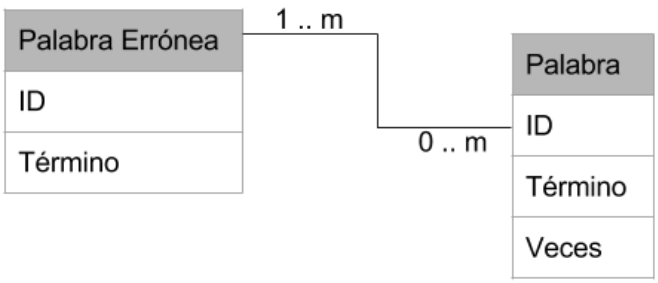

Figura 8: Modelo de la estructura de términos corregidos

Adicionalmente se maneja también el almacenamiento de la estructura de palabras ya corregidas como se observa en la Figura 8, relacionando la palabra errónea con las palabras correctas.

- Palabra errónea:
  - Término
- Palabra:
  - Término (bien escrito)
  - Veces que apareció

7.1.2 Neo4j

Para almacenar el grafo que se utilizará la base de datos Neo4j, que permite mantener esta estructura y será de fácil acceso para realizar los cambios en el componente de corrección.

Se utiliza un modelo similar a la estructura de grafo antes presentada:

- Nodo:
  - Caracter
  - Número de caracteres
  - Veces que aparecen en el diccionario
  - Relación con nodo
- Relación:
  - Valor de la unión
  - Posición

7.2. Interfaz

Se ha desarrollado una interfaz que sea fácil de usar y accesible al usuario. Se muestran las 2 pantallas principales que son: configuración, dónde se carga el diccionario con las palabras

en shipibo y que ayuda a crear las conexiones en el grafo de letras; y la del corrector, donde se muestra los campos a utilizar para realizar la corrección.

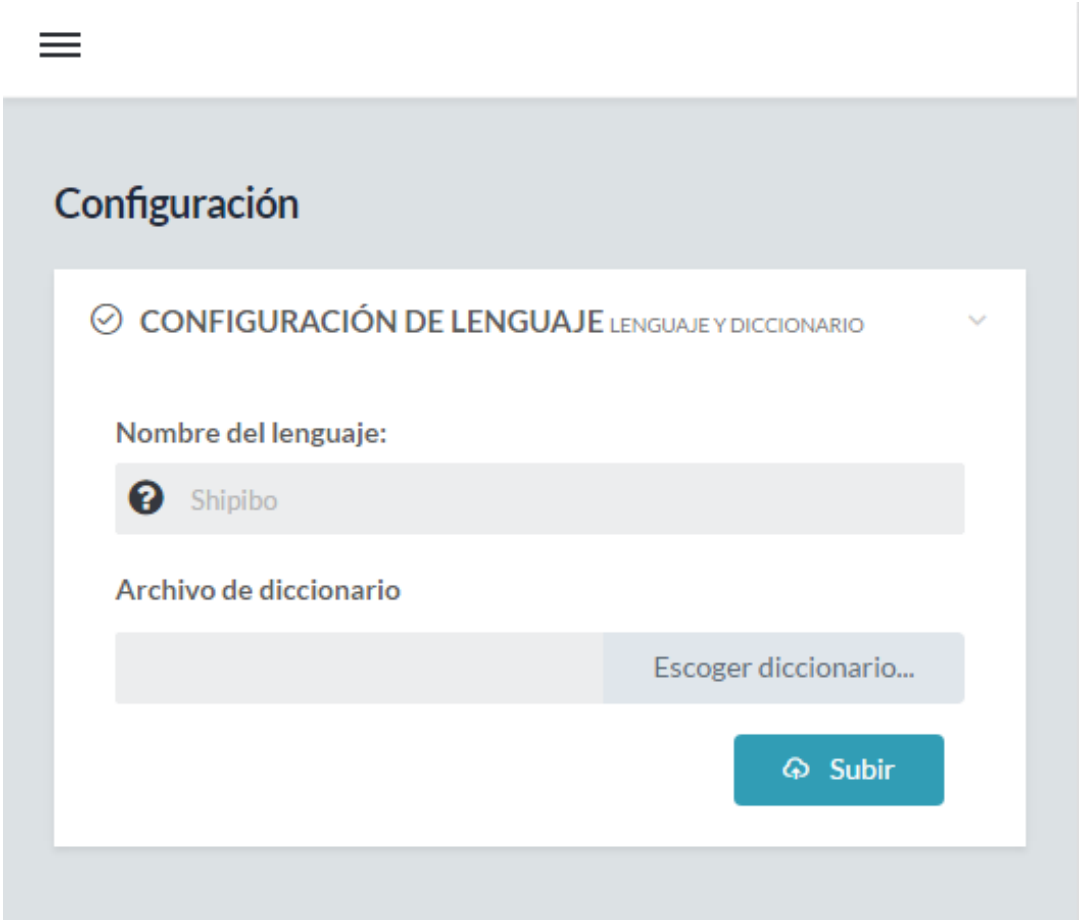

Figura 9: Pantalla de configuración

En la primera pantalla para realizar la configuración que se muestra en la Figura 9, primero se escribe la lengua que en este caso es el Shipibo-konibo. Después se da clic en "Escoger diccionario…" y se selecciona el archivo que contiene las palabras del diccionario de shipibo-konibo, este archivo debe tener formato xlsx de Microsoft Excel. Al dar clic en Subir, el archivo será leído y procesado para generar las palabras que llenarán el corpus que se utilizará, así como las estructuras de datos necesarias para la corrección.

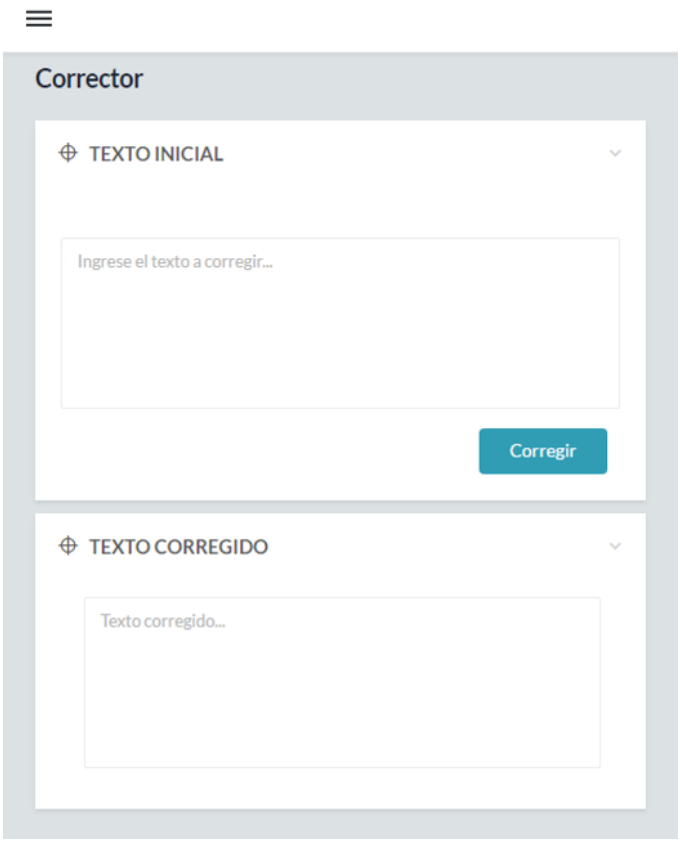

Figura 10: Pantalla de corrección

Una vez cargada la información en configuración, en la pantalla donde se muestra el corrector como se ve en la Figura 10, será donde el usuario ingrese el texto que desea corregir y selecciona el botón “Corregir”. Luego se producirá la corrección aplicando el algoritmo antes explicado y se obtendrá el resultado en el campo texto corregido con la mejor solución escogida por el algoritmo.

8. Experimentación y Resultados

Para establecer la eficacia del corrector ortográfico se realizará una experimentación utilizando métricas [van Huyssteen, et al., 2004] para evaluar este tipo de proyectos.

8.1. Diseño de experimento

Para realizar los experimentos, donde se probará la efectividad del algoritmo, se ha decidido utilizar distintas oraciones extraídas del diccionario shipibo-konibo [James, et al., 1993]. Estas oraciones corresponden a los ejemplos del uso de cada entrada del diccionario.
Como el diccionario no está con los nuevos cambios oficiales, se procedió a realizar un pre-procesamiento para actualizar todas las palabras en las oraciones de ejemplos.
Terminada la limpieza se generarán 2 tipos de pruebas. La primera prueba consiste en agregar aleatoriamente un carácter a algunas palabras de la oración. Por otro lado, la segunda prueba agrega, elimina o cambia caracteres de algunas palabras de forma aleatoria en la oración.

En una tabla se crean tres columnas: en la primera columna la oración original ya limpia, en la segunda columna la oración con el primer tipo de error y finalmente en la última columna la oración con más de 1 tipo de error en la oración. El archivo con las tablas generadas se utiliza como entrada para realizar los experimentos.
El primer experimento consiste en evaluar en base a métricas desarrolladas para correctores ortográficos [van Huyssteen, et al., 2004] y el segundo experimento consiste en conocer la clasificación de las sugerencia.

8.2. Resultados

Se realizó la corrección de los 2 tipos de prueba en 5121 oraciones. Para identificar el correcto funcionamiento se utilizaron las métricas de *Recall, Precision,* medida de sugerencias y de performance general que son propuestas para la evaluación de correctores ortográficos [van Huyssteen, et al., 2004]. Al contabilizar la cantidad de palabras en el total

de oraciones fue de 55786. Entre estas se incluyeron palabras correctas y las palabras incorrectas lo que permite evaluar mejor las métricas.

Para calcular el *recall* y *precision* de los resultados se tomó:

Verdaderos positivos (Vp): Palabras mal escritas que están bien corregidas.
Verdaderos negativos (Vn): Palabras mal escritas que están mal corregidas.
Falsos positivos (Fp): Palabras bien escritas que están bien corregidas.
Falsos negativos (Fn): Palabras bien escritas que están mal corregidas.

$$Recall = \frac{Vp}{Vp + Fp}$$

Ecuación 2: Métrica de recall

$$Precision\ = \frac{Vp}{Vp+Vn}$$

Ecuación 3: Métrica de precision

Para calcular la medida de sugerencias se utiliza un puntaje dependiendo de la sugerencia de corrección que será aplicado a todas las correcciones. Al terminar se hace una suma de todos los puntajes obtenidos de las correcciones y se divide entre la cantidad de verdaderos positivos para hallar el valor de la medida de sugerencia. Los puntajes utilizados son:

Corrección correcta en la primera posición de las sugerencias: 1 punto
Corrección correcta en alguna posición de las sugerencias: 0.5 puntos
Ninguna corrección correcta: -0.5 puntos
Sin sugerencia: 0 puntos

Para culminar con el cálculo de performance general de la corrección, se utilizará una fórmula que es:

$$Pa = \frac{Vp + Fp}{Vp + Vn + Fp + Fn}$$

Ecuación 4: Métrica de performane general

Luego de haber ejecutado el primer experimento se obtuvieron los siguientes datos:

Tabla 4: Datos del experimento tipo 1

Dato	Valor
Verdadero positivo	7099
Verdadero negativo	6276
Falso positivo	14200
Falso negativo	128
Corrección correcta en la primera posición de las sugerencias	47,24%
Corrección correcta en alguna posición de las sugerencias	28,19%
Ninguna corrección correcta	22,68%
Sin sugerencia	1,89%

Tabla 5: Datos del experimento tipo 2

Dato	Valor
Verdadero positivo	3504
Verdadero negativo	9769
Falso positivo	14242
Falso negativo	111
Corrección correcta en la primera posición de las sugerencias	44.96%
Corrección correcta en alguna posición de las sugerencias	17.89%
Ninguna corrección correcta	35.34%
Sin sugerencia	1.81%

Con estos datos que se recopilaron en la Tabla 4 del primer experimento se pudo calcular las métricas deseadas:

Tabla 6: Métricas resultantes del tipo 1

Métrica	Valor
Recall	0.33
Precision	0.53
Medida de sugerencia	14117.5 puntos
Performance general	0.76

Tabla 7: Métricas resultantes del tipo 2

Métrica	Valor
Recall	0.19
Precision	0.26
Medida de sugerencia	10230.5 puntos

Performance general	0.64

Para el segundo experimento se utilizaron los mismos datos para encontrar la clasificación de las sugerencias, para este caso se utiliza cuatro elementos: la primera posición, la top-3, la top-5 y la top-7. Esto se analiza para identificar que tan bueno es el resultado que se ofrece al usuario, los cuales se pueden ver en la tabla siguiente.

Tabla 8: *Top* de palabras en las sugerencias

Top	Prueba tipo 1	Prueba tipo 2
1	62.74%	71.54%
3	3.38%	2.12%
5	3.11%	1.90%
7	30.77%	24.44%

8.3. Discusión

Con los resultados obtenidos que se ven en la Tabla 4 y Tabla 6, los valores de las métricas de *Recall* y *precision* son bajos, sin embargo esto se debe a que el corrector aún tiene que ser mejorado para identificar mejor los errores. Lo que haría mejorar su eficacia es poder detectar de mejor manera las palabras que no necesitan ser realmente corregidas, ya que como se puede apreciar, muchas de las palabras que se corrigen son palabras que no lo necesitan. Para atacar ese problema se pueden utilizar distintos enfoques. Uno de ellos es aprovechar el corpus disponible para realizar una búsqueda de la palabra e identificar en el corpus si ya existe. Así se evitaría en mayor medida evitar una corrección innecesaria, sin embargo supondría un aumento en el tiempo de corrección debido a que se realiza una búsqueda por cada palabra que si bien en textos cortos no sería mucha la diferencia, en textos más extensos sí se notaría. Para tomar la decisión de incluir este enfoque se debería implementarlo y realizar las pruebas nuevamente.

A pesar de los bajos números en *recall* y *precision*, se obtiene un buen resultado a nivel general en base a las métricas para evaluar correctores ortográficos [van Huyssteen, et al., 2004], y esto se debe a que se considera dentro del resultado correcto que tanto palabras mal escritas como bien escritas al ser corregidas ofrecen un resultado positivo. Además, considerando los datos obtenidos de las correcciones se puede apreciar que más de la mitad de las veces se ha encontrado una propuesta de corrección correcta, y el 25% de las veces estas correcciones están en la primera posición como sugerencia en los dos tipos de pruebas realizadas. Asimismo, vemos que cuando se enfoca sólo en el tipo de error de inserción, se obtienen mejores resultados que al afrontar a todos los tipos de error (inserción, sustracción e intercambio de caracteres).

Adicionalmente, respecto al experimento para hallar en qué posición se encuentra la solución correcta dentro de las sugerencias, se aprecia que más de la mitad de palabras corregidas se encuentran en la primera posición de las sugerencias de solución del corrector, demostrando nuevamente su eficacia.

9. Conclusiones y trabajos futuros

Al iniciar con la investigación para elaborar el proyecto de tesis se conoció más acerca de los problemas que los hablantes de lenguas originarias tenían, además de conocer más acerca de las iniciativas que tiene el gobierno para ser más inclusivos. Como ayuda con la inclusión, el corrector ortográfico es un buen recurso que podrá ser aprovechado por los hablantes para mejorar su ortografía. Adicionalmente dio pie a publicar un *paper*[Alva, C. , et al., 2017] en una conferencia relacionada a la investigación en procesamiento de lenguaje natural donde se dio a conocer la iniciativa de este proyecto.

A lo largo del desarrollo del proyecto se pudo investigar distintos enfoques para el desarrollo de un corrector ortográfico tales como basado en corpus, basado en reglas, basados en análisis de contexto y enfoques estadísticos. Pero como elección, se utilizó un híbrido con ayuda del corpus y en base a reglas de ortografía extraídas del texto oficial del shipibo-konibo. Sin embargo, conforme se iba desarrollando para mantener la estructura de los grafos que son base en el algoritmo, se encontró una base de datos especializada en el almacenamiento de estas estructuras como neo4j que solucionó el inconveniente del mantener los vertices y sus relaciones.

Otra dificultad que se encontró fue el utilizar la recursividad a la hora de seguir distintos caminos al encontrar un error en las palabras, ya que se podía cambiar la letra, agregarla o eliminarla. Por eso se definió un máximo de profundidad que ayudó a que no se creen caminos tan largos.

Finalmente se obtuvieron los resultados, los cuales no fueron absolutamente buenos, porque el enfoque seguido no es suficiente para obtener una corrección precisa y veloz. Sin embargo, se apreciaron resultados parciales prometedores respecto al error de inserción, además de que

la sugerencia de las posibles correcciones suele presentar el resultado correcto entre sus primeras opciones.

Como trabajos futuros puede agregarse un análisis de contexto para poder identificar mejor las palabras que no deben ser corregidas, debido a que estas pueden estar bien escritas pero no ser adecuadas para el contexto de la oración. Además, se puede implementar un algoritmo de corrección con un enfoque iterativo que podría mejorar la velocidad de procesamiento al realizar la corrección. De este modo, podría pasar de ser una aplicación web a una aplicación móvil, ya que actualmente existe una tendencia hacia el desarrollo de aplicativos móviles.

Otra tendencia que tiene buenos resultados es la inteligencia artificial y el aprendizaje de máquina por lo que se podría utilizar redes neuronales para la identificación de palabras más escritas y su posible solución, podría acercarnos a una eficiencia como un autocorrector de Google.

Bibliografía

Abdullah, A. B. A., & Rahman, A. (2003, November). A Generic Spell Checker Engine for South Asian Languages. In Conference on Software Engineering and Applications (SEA 2003) (pp. 3-5).

Aduriz, I., Urkia, Miriam. I. R. I. A. M., Alegria, I. N. A. K. I., Artola, X. A. B. I. E. R., Ezeiza, N. E. R. E. A., & Sarasola, K. E. P. A. (1997). A spelling corrector for Basque based on morphology. Literary and Linguistic Computing, 12(1), 31-38.

Aikman, S. (1999). Intercultural education and literacy: An ethnographic study of indigenous knowledge and learning in the Peruvian Amazon (Vol. 7). John Benjamins Publishing.

Allardt, E. (1984). What constitutes a language minority? Journal of Multilingual & Multicultural Development, 5(3-4), 195-205.

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). Web services. In Web Services (pp. 123-149). Springer Berlin Heidelberg.

Alva, C. A. y Oncevay, F. A. (2017). Spell-checking based on syllabification and character-level graphs for a Peruvian agglutinative language. En EMNLP 2017 Workshop on Subword and Character Level Models in NLP, SCLeM 2017. ACL Anthology. Recuperado de: https://aclanthology.info/papers/W17-4116/w17-4116

Aronoff, M. (1994). Morphology by itself: Stems and inflectional classes (No. 22). MIT press.

Barari, L., & QasemiZadeh, B. (2005). CloniZER spell checker adaptive language independent spell checker. In AIML 2005 Conference CICC, Cairo, Egypt (pp. 19-21).

Barth, F. (1976). Los grupos étnicos y sus fronteras (Vol. 54). México: Fondo de cultura económica.

Booij, G. (2012). The grammar of words: An introduction to linguistic morphology. Oxford University Press.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. Communications of the ACM, 7(3), 171-176.

Deza, M. M., & Deza, E. (2009). Encyclopedia of distances. In Encyclopedia of Distances (pp. 1-583). Springer Berlin Heidelberg.

Diaconescu, S., Ingineru, C., Codirlasu, F., Rizea, M., & Bulibasa, O. (2009, June). General system for normal and phonetic inflection. In Speech Technology and Human-Computer Dialogue, 2009. SpeD'09. Proceedings of the 5-th Conference on (pp. 1-10). IEEE.

Elhuyar Hizkuntza eta Teknologia, I. (2016). Xuxen.eus - Xuxen - Xuxen: página web del corrector ortográfico y gramatical para el euskera. [Online] Xuxen.eus. Consultado el 12 de septiembre del 2016, desde http://xuxen.eus/.

Faust, N. (1973). Lecciones para el aprendizaje del idioma shipibo-conibo. Documento de trabajo, 1.

Hartmann, W., Le, V. B., Messaoudi, A., Lamel, L., & Gauvain, J. L. (2014). Comparing decoding strategies for subword-based keyword spotting in low-resourced languages. In INTERSPEECH (pp. 2764-2768).

James, L., Lauriault, E., & Day, D. (1993). Diccionario Shipibo-Castellano. Lima: Instituto Lingüístico de Verano.

Jara Males, P., & Gonzalez Acero, C. (2015). Estrategias institucionales y modalidades de atención en servicios para la inclusión social de poblaciones vulnerables. Inter-American Development Bank.

Jurafsky, D., & Martin, J. H. (2014). Speech and language processing. Pearson.

Lee, L. F. (1997). A smooth likelihood simulator for dynamic disequilibrium models. Journal of econometrics, 78(2), 257-294.

Ministerio de educación del Perú (2013). Encuesta nacional de tecnologías de y comunicación. Consultado el 28 de abril del 2017, http://educaciontic.perueduca.pe/?p=810

Morin, F. (1998). Los shipibo-conibo. Guía etnográfica de la Alta Amazonía, 3, 275-435.

Mysql.com. (2017). MySQL. Consultado el 15 de marzo del 2017, desde https://www.mysql.com/.

Natural Language Toolkit — NLTK 3.0 documentation. (2016). Nltk.org. Consultado el 26 de octubre del 2016, desde http://www.nltk.org/.

Ndiaye, M., & Faltin, A. V. (2003). A spell checker tailored to language learners. Computer Assisted Language Learning, 16(2-3), 213-232.

Neo4j Graph Database. (2017). Neo4j, the world's leading graph database - Neo4j Graph Database. Consultado el 15 de marzo del 2017, desde https://neo4j.com/.

Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.

Przybocki, M., Sanders, G., & Le, A. (2006). Edit distance: a metric for Machine Translation evaluation. In LREC-2006: Fifth International Conference on Language Resources and Evaluation (pp. 2038-2043).

Real Academia Española. (2014). Lengua. En Diccionario de la lengua española (23.a ed.). Recuperado de: http://dle.rae.es/?id=N77BOIl

Real Academia Española. (2014). Lingüistica. En Diccionario de la lengua española (23.a ed.). Recuperado de: http://dle.rae.es/?id=NNPFPOl

Real Academia Española. (2014). Ortografía. En Diccionario de la lengua española (23.a ed.). Recuperado de: http://dle.rae.es/?id=RG9EvWw

Raghavan, V., Bollmann, P., & Jung, G. S. (1989). A critical investigation of recall and precision as measures of retrieval system performance. ACM Transactions on Information Systems (TOIS), 7(3), 205-229.

Riehle, D. (2000). Framework design (Doctoral dissertation, Diss. Technische Wissenschaften ETH Zürich, Nr. 13509, 2000).

Rivera, A. C. (2001). Atlas lingüístico del Perú (Vol. 6). Centro Bartolomé de Las Casas.

Rodríguez-Ortega, D. (2015). Un bien necesario para la escritura: la competencia ortográfica. Ocnos: Revista de estudios sobre lectura, (13), 85-98.

Rolando, G. (2012). Una mirada al Estado desde la educación en una comunidad shipibo-conibo: education in a Shipibo-Conibo community. Anthropologica, 30(30), 45-76.

Sapir, E. (1966). El lenguaje. Fondo de cultura económica.

Schwaber, K., & Beedle, M. (2002). Agile software development with Scrum (Vol. 1). Upper Saddle River: Prentice Hall.

Sow, D., & Eleftheriadis, A. (1998, November). Representing information with computational resource bounds. In Signals, Systems & Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on (Vol. 1, pp. 452-456). IEEE.

Sullón Acosta, K. N., Huamancayo Curi, E., Mori Clement, M., & Carbajal Solis, V. (2013). Documento nacional de lenguas originarias del Perú.

The Web framework for perfectionists with deadlines | Django. (2016). Djangoproject.com. Consultado el 26 de octubre del 2016, desde https://www.djangoproject.com/

Tolentino, H. D., Matters, M. D., Walop, W., Law, B., Tong, W., Liu, F., ... & Payne, D. C. (2007). A UMLS-based spell checker for natural language processing in vaccine safety. BMC medical informatics and decision making,7(1), 1.

Van Huyssteen, G. B., Eiselen, E. R., & Puttkammer, M. J. (2004). Re-evaluating evaluation metrics for spelling checker evaluations. In Proceedings of First Workshop on International Proofing Tools and Language Technologies (pp. 91-99).

Vansina, J., & Udina, D. (2007). Tradición oral, historia oral: Logros y perspectivas (Oral tradition, oral history: Achievements and perspectives).Historia, Antropología y fuentes orales, 151-163.

Vásquez, A. C., Quispe, J. P., & Huayna, A. M. (2009). Procesamiento de lenguaje natural. Revista de investigación de Sistemas e Informática, 6(2), 45-54.

Vásquez Gil, S. M. (2016). Concepciones de acompañantes pedagógicos de escuelas EIB shipibo-konibo sobre el acompañamiento pedagógico. 2016.

Vigil, N. (2005). Pueblos indígenas y escritura. Portal aula intercultural, http://aulaintercultural.org/2005/02/25/pueblos-indigenas-y-escritura/. Consultado el, 28.

Ward, M. (2014). Using Irish NLP resources in primary school education. CLTW 2014, 6.

Wasala, R. A., Weerasinghe, R., Pushpananda, R., Liyanage, C., & Jayalatharachchi, E. (2011). An Open-Source Data Driven Spell Checker for Sinhala. ICTer, 3(1).

Welcome to Python.org. (2016). Python.org. Consultado el 26 de octubre del 2016, desde https://www.python.org/