DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: CREACIÓN DE UNA BASE DE DATOS PARA LA EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE RECONOCIMIENTO DE FÓRMULAS MATEMÁTICAS EXTRAIDAS DESDE UN ARCHIVO EN FORMATO PDF.

AUTOR: BACA ESPINOSA, MICHAEL RENATO

DIRECTOR: Ing. LARCO BRAVO, JULIO CESAR

SANGOLQUÍ

2018

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación "**CREACIÓN DE UNA BASE DE DATOS PARA LA EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE RECONOCIMIENTO DE FÓRMULAS MATEMÁTICAS EXTRAIDAS DESDE UN ARCHIVO EN FORMATO PDF**" realizado por el señor **MICHAEL RENATO BACA ESPINOSA,** ha sido revisado en su totalidad y analizado por software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE, por lo tanto me permito acreditarlo y autorizar al señor **MICHAEL RENATO BACA ESPINOSA** para que lo sustente públicamente.

Sangolquí, 30 de enero del 2018

Ing. Julio Cesar Larco Bravo
Director del proyecto

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

Yo, **MICHAEL RENATO BACA ESPINOSA** con cédula de identidad N°1721739116, declaro que este trabajo de titulación "**CREACION DE UNA BASE DE DATOS PARA LA EVALUACION DEL DESEMPEÑO DE ALGORITMOS DE RECONOCIMIENTO DE FORMULAS MATEMATICAS EXTRAIDAS DESDE UN ARCHIVO EN FORMATO PDF**" ha sido desarrollada considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente Declaramos que este trabajo es de mi autoría, en virtud de ello me declaro responsable del contenido, veracidad y alcance de la investigación mencionada.

Sangolquí, 30 de enero del 2018

Michael Renato Baca Espinosa
C.I. 1721739116

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **MICHAEL RENATO BACA ESPINOSA**, autorizo a la Universidad de las Fuerzas Armadas-ESPE publicar en la biblioteca virtual de la institución el presente trabajo de titulación "**CREACION DE UNA BASE DE DATOS PARA LA EVALUACION DEL DESEMPEÑO DE ALGORITMOS DE RECONOCIMIENTO DE FORMULAS MATEMATICAS EXTRAIDAS DESDE UN ARCHIVO EN FORMATO PDF**" cuyo contenido, ideas y criterios son de mi autoría y responsabilidad.

Sangolquí, 30 de enero del 2018

Michael Renato Baca Espinosa
C.I. 1721739116

DEDICATORIA

A Dios, por permitirme un día más de vida, y salud que me han ayudado para que poco a poco logre objetivos de vida profesionales y personales.

A mis padres Renato y Sonia, por su infinito amor y apoyo incondicional en todas las etapas de mi vida, sus consejos, constantes motivaciones pero sobre todo su gran paciencia y ejemplo para inculcarme todos los valores necesarios para ser una persona de bien y nunca claudicar ante las adversidades.

A mi segunda madre Liria, por su constante apoyo, amor inigualable y nunca faltarme en todas las etapas de mi vida, por siempre estar a mi lado inculcándome valores y enseñándome a siempre cumplir con mis responsabilidades y mediante su gran ejemplo siempre amar a Dios.

A mis Hermanos Diana y Francisco, por siempre estar a mi lado en las buenas y en las malas dándome su apoyo, los amo ñaños.

A mis amigos y compañeros, por todos los momentos difíciles vividos, siempre apoyándonos y dándonos palabras de aliento para no desmayar en este largo camino de grandes logros. En especial a Marcela por su compañía en gran parte de este sendero que juntos nos apoyamos incondicionalmente, sé que no cambiará.

A todas las personas que de una u otra manera han formado parte de mi vida, no me queda más que agradecerles su apoyo, esto también es gracias a ustedes.

Michael Baca Espinosa.

AGRADECIMIENTO

Doy gracias a Dios por permitirme culminar una etapa más de mi vida junto a mi familia y personas que amo, por la salud que nos ha dado y las infinitas bendiciones que han colmado mi vida.

A toda mi familia por su constante apoyo y siempre velar por mi bienestar. Renato, Sonia, Liria, Romeo, Diana, Francisco y familia en general, gracias a todos ustedes, sus consejos, y palabras de aliento han sido indispensables en toda esta etapa que ahora llega a su fin.

A mis amigos que mediante su apoyo y colaboración siempre nos brindamos ayuda en los momentos más difíciles, por compartir grandes experiencias y anécdotas de vida que nunca olvidaremos.

A los docentes de la universidad que han sabido brindar y transmitir su conocimiento sin ninguna clase de egoísmo, muchas gracias, sus enseñanzas me ayudarán a llegar lejos y quedarán en mí para toda la vida, una mención especial al ingeniero Julio Larco ya que gracias a sus sólidos conocimientos, su dirección y apoyo pude lograr este trabajo.

A la Universidad de las Fuerzas Armadas – ESPE, por permitirme formar parte de esta prestigiosa familia que conforman sus estudiantes y profesionales, al departamento CIRAD ya que gran parte de sus integrantes han sido parte fundamental de mi formación profesional.

Gracias a todas las personas que con su apoyo y conocimiento fueron parte de este trabajo.

Michael Baca Espinosa.

Tabla de contenido

Índice de Figuras

Índice de Tablas

Resumen

En la actualidad el paso de información científica en su mayoría se da mediante artículos que en gran parte de los casos se encuentran en formato PDF, lo cual ha hecho que crezca la popularidad de dicho formato y que hace necesario manipular este tipo de documentos, tareas como extraer texto, tablas, figuras y formulas son ineludibles para ser analizadas y procesadas. Una de las tareas más importantes en la detección y reconocimiento de fórmulas matemáticas es identificar correctamente su ubicación dentro de un documento, uno de los principales problemas en todos estos trabajos dedicados a la detección de fórmulas es validar su desempeño ya que los programas y las bases de datos con las que se puede realizar la validación no son validas o no son de uso libre. En este proyecto se busca mediante el procesamiento de archivos pdf y procesamiento digital de imágenes crear una base de datos que contenga posición y caracteres de fórmulas matemáticas extraídas de un archivo en formato PDF. Para eso se usará como base el trabajo propuesto por (Xiaoyan Lin L. G., 2012) para que nuevos algoritmos y los ya existentes de reconocimiento posición y caracteres de fórmulas matemáticas puedan ser evaluados o probados para tener un criterio equitativo de rendimiento.

Palabras Clave:

- **DETECCIÓN DE FÓRMULAS MATEMÁTICAS.**
- ***BOUNDING BOX (BBOX.)***
- **HERRAMIENTAS DE EVALUACIÓN.**
- **PROCESAMIENTO DIGITAD DE IMÁGENES (PDI).**
- **COMPONENTES CONEXOS.**

Abstract

At present, the passage of scientific information is mostly through articles that in many cases are in PDF format, which has made the popularity of this type of format grow and makes it necessary to manipulate this type of documents, tasks such as extract text, tables, figures and formulas are inescapable to be analyzed and processed. One of the tasks that are pending in terms of working with files of this type is the mathematical formula identification. One of the most important tasks in the detection and recognition of mathematical formulas is to correctly identify their location within a document, the main problems in all these works dedicated to the detection of formulas is to validate their performance because the programs and databases that they can usea are not meaningful or are not free to use. In this project with the processing of pdf files and digital image processing, we will create a database that contains a position of formulas and characters extracted from a PDF file. For this, the work proposed by (Xiaoyan Lin L. G., 2012) be used as a basis. For the evaluation of performance a metric will be proposed for the mathematical formula identification.

Keywords:

- **MATHEMATICAL FORMULA IDENTIFICATION.**
- **BOUNDING BOX (BBOX.)**
- **EVALUATION TOOLS.**
- **DIGITAL IMAGE PROCESSING.**
- **RELATED COMPONENTS.**

CAPITULO 1

Generalidades

1.1. Introducción

En la actualidad el intercambio de información científica generalmente se da mediante artículos científicos que en la mayoría de los casos se encuentran en formato PDF, siglas de *Portable Document Format*, lo cual ha hecho crecer la popularidad de dicho formato haciendo necesario su manipulación y la extracción de información en este tipo de formato. Tareas como: extraer texto, tablas, figuras y fórmulas son ineludibles para ser analizadas y procesadas. De estas tareas una que se encuentra pendiente en cuanto al trabajar con archivos de este tipo es el reconocimiento de fórmulas matemáticas. La extracción de información de archivos en formatos PDF es importante y de gran ayuda para escaneo de archivos, conversión de texto a PDF y tecnologías asistivas destinadas a la ayuda de personas con deficiencias.

Uno de los fines con los que se realiza este tipo de investigaciones y análisis es para ayudar a desarrollar herramientas que faciliten la lectura y aprendizaje de personas con discapacidad visual y de esta forma contribuir con lo que dice la Constitución de la República del Ecuador en su título II, capítulo tercero, sección sexta, artículo 47, apartado 7, donde se reconoce a las personas con discapacidad, el derecho a: “Una educación que desarrolle sus

potencialidades y habilidades para su integración y participación en igualdad de condiciones. Se garantizará su educación dentro de la educación regular. Los planteles regulares incorporarán trato diferenciado y los de atención especial la educación especializada. Los establecimientos educativos cumplirán normas de accesibilidad para personas con discapacidad e implementarán un sistema de becas que responda a las condiciones económicas de este grupo." (Asamblea Constituyente, 2008).

Uno de los problemas que encuentran los desarrolladores de herramientas para facilitar la lectura y aprendizaje de personas con discapacidad visual, específicamente en el área de reconocimiento de fórmulas matemáticas en archivos con formato PDF es la falta de bases de datos válidas, es decir con una amplia información, para probar o evaluar el desempeño de su trabajo frente a otros, ya que en la literatura se pueden encontrar bases de datos pero estas no están disponibles libremente.

En este proyecto se busca mediante el procesamiento de archivos pdf y procesamiento digital de imágenes crear una base de datos que contenga posición y caracteres de fórmulas matemáticas extraídas de un archivo en formato pdf, utilizando Python y OpenCV (*Open-source Computer Vision*). Para esto se usará como base el trabajo propuesto por Xioyan Lin (Xiaoyan Lin L. G., 2012) para que nuevos algoritmos y los ya existentes de reconocimiento de posición y caracteres de fórmulas matemáticas puedan ser evaluados o probados para tener un criterio equitativo de rendimiento.

1.2. Justificación e Importancia

Las matemáticas son aplicadas extensamente en varios ámbitos como la educación, investigación, negocios, áreas técnicas y administrativas, y en otros campos del conocimiento, convirtiéndose por lo tanto en un lenguaje universal sin importar el área de estudio o aplicación de las mismas, siendo exactas para ayudar a comprender diferentes fenómenos del diario vivir. Por lo tanto desarrollo de herramientas que faciliten el aprendizaje a personas con discapacidad visual, es un área importante de investigación una de estas áreas es el reconocimiento de fórmulas matemáticas en archivos con formato PDF. Es por esto y más que el procesamiento automático para la extracción de fórmulas matemáticas de documentos es realmente necesario e importante.

Existen varios algoritmos que extraen texto de archivos PDF, pero hay pocos que extraen información de archivos con formato PDF cuando se trata de fórmulas matemáticas provocando el inconveniente de que no se puede evaluar su desempeño debido a que las bases de datos existentes no son de uso libre o no son realmente válidas como para evaluar el desempeño.

El principal problema que tienen los creadores de algoritmos de identificación de fórmulas matemáticas es la validación de su trabajo debido a que cada uno de ellos crea su

propia base de datos y muchas veces esta no está disponible lo que hace difícil hacer una comparación del desempeño de los diferentes algoritmos creados.

Una solución parcial a este problema es el trabajo realizado por Xiaoyan Lin (Xiaoyan Lin L. G., 2012) en el que se propone una base de datos para poder evaluar el rendimiento de programas de identificación de fórmulas matemáticas en archivos con formato pdf, esta base de datos está compuesta por el análisis a páginas de documentos que se recogen de diferentes tipos de documentos, incluyendo revistas, actas de conferencias, libros, informes técnicos, etc. La distribución de cada tipo de fuente se muestra en la Tabla 1 *Distribución* de fuentes.

Tabla 1

Distribución de fuentes.

Fuente	Conferencias	Revistas	Libros	Informes	Otros	Total
Número	93	73	7	16	5	194

Fuente: (Xiaoyan Lin L. G., 2012)

El inconveniente es que solo logra evaluar la posición de las fórmulas y diferentes tipos de identificación errónea pero no permite evaluar los elementos que forman parte de la fórmula lo cual denota un problema para los algoritmos que deseen evaluar su rendimiento en identificación de elementos dentro de una fórmula matemática.

Por lo tanto, en el reconocimiento de fórmulas específicamente en documentos PDF sería de gran utilidad para la comunidad de investigadores que trabajan en esta área la existencia de una base de datos abierta y gratuita formada por fórmulas aisladas y fórmulas incrustadas en el texto, de tal manera que al contar con esta base de datos y una métrica de evaluación permita evaluar el desempeño de algoritmos de reconocimiento de fórmulas matemáticas.

Con este tipo de análisis y aportes, se contribuye a la investigación de herramientas con mejor calidad que podrán ser utilizadas para ayudar a personas con discapacidad visual a tener una mejora en su estilo de vida principalmente contribuyendo en facilitar su desenvolvimiento en el área de la educación, de esta manera haciendo respetar su derecho a una educación inclusiva la cual está descrita en la Ley Orgánica de Discapacidades (Asamblea Nacional del Ecuador, 2012) en su capítulo segundo, sección tercera, Art.28

1.3. Estado del Arte

La extracción de información de documentos viene de la necesidad de sintetizar la gran cantidad de información que existe actualmente, se empezó con el análisis de imágenes de documentos donde se extrae información a partir de características de bajo nivel, características de forma, información lingüística, entre otros, este tipo de técnica se ha aplicado en trabajos (Garain, 2009) donde se han obtenido muy buenos resultados, que se han podido evaluar mediante bases de datos publicadas para facilidad de este tipo de técnicas basadas en imágenes (UW-III, 1996), con el inconveniente de que dicha base de datos solo

presenta 25 páginas de documentos conteniendo 100 fórmulas matemáticas, está claro que la base de datos es muy pequeña como para ser lo bastante representativa en casos prácticos.

El análisis de documentos escaneados es otra etapa donde a lo largo de los años se han obtenido buenos resultados en varios trabajos realizados, uno de los métodos que se ha propuesto es el de inicialmente invocar un OCR (*Optical Character Recognition*) como reconocedor del documento de entrada, con este método se miden los niveles de palabra de cada línea y un análisis estadístico muestra que dichos niveles son diferentes en oraciones con expresiones matemáticas incrustadas que en las que no tienen expresiones matemáticas, este método se describe en el trabajo propuesto *Identification of Embedded Mathematical Expressions in Scanned Documents* (Utpal Garain, 2004), donde también se propone una métrica de evaluación para poner a prueba algoritmos y evaluar los diferentes resultados obtenidos según sus propios parámetros que se muestran en la Figura 1.

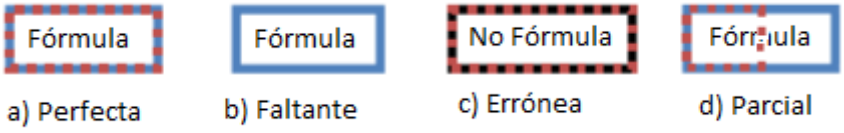


Figura 1. Parámetros para evaluación de reconocimiento de fórmulas en *Identification of Embedded Mathematical Expressions in Scanned Documents*

Fuente*: (Utpal Garain, 2004)*

- Perfectos: Cuando las coordenadas extraídas contienen perfectamente toda la fórmula matemática.
- Faltantes: No se extraen coordenadas de fórmulas que si existen en el documento.
- Erróneos: Se extraen coordenadas donde no existe ninguna fórmula matemática.
- Parciales: Al extraer las coordenadas de una fórmula, estas no la contienen en su plenitud a dicha fórmula.

El reconocimiento de expresiones matemáticas en documentos PDF es un área relativamente nueva e importante en el campo del análisis de este tipo de documentos ya que es diferente de los métodos de extracción de fórmulas matemáticas basadas en imágenes de documentos, varios de los algoritmos existentes en la literatura proponen métodos basados en reglas (*rule-based*) y basados en aprendizaje (*learning-based*) para el reconocimiento de fórmulas aisladas y embebidas, este método es usado en el trabajo realizado por Lin (Xiaoyan Lin L. G., 2011). Otro del os métodos usados es basándose en la ventana de Parzen como indica Jianming Jin en su trabajo (Jianming Jin, 2003). Una de las ventajas de trabajar directamente con documentos PDF es que se puede tener conocimiento perfecto de los caracteres utilizados en las fórmulas, extraídos directamente del documento por lo cual existen trabajos realizados en esta área (Josef B. Baker, 2009), (Xiaoyan Lin L. G., 2013).

Para evaluar la efectividad y validez de los trabajos mencionados anteriormente se ha usado bases de datos ya sean propias de cada autor o públicas como en el trabajo publicado por M. Suzuki (M. Suzuki, 2005) donde presenta una base de datos para la evaluación de

rendimiento de reconocimiento de fórmulas matemáticas pero con el inconveniente de que las áreas de las fórmulas no se entregan directamente. Se ha presentado grandes bases de datos que se consideran validas (U. Garain, 2005) (K. Ashida, 2006) ya que tienen un gran contenido de información, pero que lamentablemente no son públicas lo que genera un inconveniente para los autores que deseen evaluar el rendimiento de sus trabajos.

Sin embargo, en el trabajo realizado por Xiaoyan Lin (Xiaoyan Lin L. G., 2012) se publica gratuitamente, con fines educativos e investigativos, una base de datos con un contenido amplio, que puede considerarse representativo para casos prácticos, con la que se puede evaluar el desempeño de reconocimiento de fórmulas matemáticas, y se propone una métrica de evaluación tomando en cuenta diferentes tipos de resultados obtenidos con un mayor análisis que en el trabajo mencionado anteriormente (Utpal Garain, 2004) permitiendo una evaluación con mayor efectividad, pero esta base de datos aún contiene errores en la codificación de los caracteres exactos que se encuentran dentro de una fórmula, así también excluye de la codificación algunos caracteres, por lo tanto en el caso de que se desee utilizar esta base de datos para evaluar el desempeño de algoritmos que, a más de reconocer la posición de fórmulas matemáticas también reconozcan la posición de todos los caracteres dentro de la mismo y el tipo de caracter reconocido, se presentarían resultados que no son reales debido a los errores presentes en la base de datos en estos campos.

1.4. Alcance

El presente proyecto se centra en extraer la información existente en la base de datos (Xiaoyan Lin L. G., 2012) para su análisis y posteriormente realizar un programa en Python, que es un lenguaje de programación de alto nivel orientado a objetos. Mediante este programa se creará una base de datos que contenga la información necesaria para poder evaluar algoritmos de reconocimiento de fórmulas matemáticas y además algoritmos que reconozcan los elementos dentro de las fórmulas. Este proceso estará basado en el procesamiento digital de imágenes; el programa que se realizará en este proyecto permitirá: corregir la codificación de caracteres dentro de la fórmula en caso de ser necesario, ingresar caracteres que no estén codificados dentro de la fórmula, todo esto se podrá realizar mediante una interfaz gráfica.

El siguiente paso una vez validada la base de datos es proponer una métrica de evaluación de desempeño para la identificación de caracteres dentro de fórmulas matemáticas tomando en cuenta todos los diferentes resultados obtenidos y los distintos criterios expuestos por diferentes autores que se mencionaron anteriormente, esta evaluación se la realizará mediante una herramienta automática que, mediante el uso de la base de datos que se realizará previamente, evaluará el rendimiento de algoritmos en ésta que, a más de reconocer fórmulas matemáticas, también reconozca los caracteres dentro de las fórmulas.

Para probar los resultados se someterá a un algoritmo existente en la literatura a la herramienta de evaluación automática con el fin de comprobar su funcionalidad y obtener un criterio equitativo de rendimiento.

1.5. Objetivos

1.5.1. General

Crear una base de datos para el reconocimiento de fórmulas matemáticas extraídas desde un archivo en formato PDF.

1.5.2. Específicos

- Investigar el estado del arte sobre sistemas de reconocimiento de fórmulas matemáticas y las bases de datos creadas para su evaluación.
- Implementar un programa para la extracción de información de la base de datos (Xiaoyan Lin L. G., 2012).
- Aplicar técnicas de procesamiento digital de imágenes para la extracción de información de la imagen de fórmulas matemáticas obtenida de un archivo PDF.
- Proponer una métrica de evaluación del desempeño para la identificación de fórmulas matemáticas.
- Crear una base de datos con los resultados obtenidos para fines académicos e investigativos.
- Probar la validez de la base de datos obtenida mediante un algoritmo de la literatura existente.

CAPITULO 2

Marco Teórico

En el siguiente capítulo se dará una descripción de la teoría necesaria para el entendimiento de como se desarrollará el proyecto. En primera instancia se explica los métodos de procesamiento digital de imágenes que son usados para el reconocimiento de fórmulas matemáticas y también se explicará a detalle el método de procesamiento digital de imagenes usado en el presente trabajo. De igual manera se expondrá un breve análisis de los archivos en formato PDF y los métodos de reconocimiento de fórmulas matemáticas usados paras este tipo de formato. Finalmente sedará a conocer la importancia de este trabajo para estudios futuros que ayuden a la inclusión de personas con discapacidad visual.

2.1. Procesamiento Digital de Imágenes

En los últimos años el procesamiento digital de imágenes ha adquirido una gran importancia en las tecnologías de la información, así también en distintos campos de estudio ya que es fundamental para varios tipos de aplicaciones como por ejemplo en la medicina, percepción remota, exploración espacial, biometría, robótica entre otras. Una de las ventajas en cuanto a la reducción del costo computacional es que actualmente se puede utilizar un computador personal para realizar algunos tipos de procesamiento digital de imágenes.

El procesamiento digital de imágenes se lo viene estudiando y experimentando desde los años 1920-1930, cuando se mejoró las imágenes digitalizadas de un periódico para su posterior transmisión entre Londres y Nueva York mediante el uso de un cable submarino (Torres, 2015).

De los principales inconvenientes que se tuvo en el inicio del estudio de estos procesos fue el elevado costo computacional que requería para realizar un óptimo trabajo y que en aquellos tiempos no se contaba con la tecnología necesaria, por esta razón con el aparecimiento de las computadoras digitales y su mejorado y veloz procesamiento de tareas se pudo divisar el verdadero potencial que tenía la manipulación de imágenes mediante la digitalización.

Desde 1964 se ha realizado el procesamiento digital de imágenes mediante el uso de computadoras, en sus inicios fueron implementados en las misiones de la NASA en viajes a la Luna para transmitir imágenes tomadas de la superficie lunar por el “Ranger 7”, de ahí en adelante todos los avances de la ciencia en el campo del procesamiento digital de imágenes ha ayudado a mejorar sustancialmente el estilo de vida de las personas ya que mediante las diferentes aplicaciones que se le puede dar en los diferentes campos de estudio, como ya se mencionó anteriormente, ha ayudado por ejemplo a los profesionales de la medicina a diagnosticar y tratar de manera más eficiente a sus pacientes.

Otro de los campos en el que el procesamiento digital de imágenes ha tenido un gran impacto es en la ayuda de aplicaciones para personas no videntes o con algún tipo de discapacidad visual pues existen métodos por los cuales se puede analizar documentos digitalizados para extraer texto, imágenes e incluso fórmulas matemáticas y de esta forma realizar diferentes técnicas para ayudar a la lectura, comprensión y así facilitar el aprendizaje a estas personas que requieren de herramientas especiales para conocer el contenido de un documento digitalizado.

Se conoce como Procesamiento Digital de Imágenes (PDI), "Al conjunto de técnicas y procesos para descubrir o hacer resaltar información contenida en una imagen usando como herramienta principal una computadora" (Torres, 2015), se puede mencionar muchas finalidades de implementar técnicas de PDI a una imagen pero principalmente es la aplicación que se le dé a la misma la que tiene mayor importancia pues, como se mencionó anteriormente, este proceso ayuda a analizar, deducir y tomar decisiones indistintamente del área en la que se las aplique.

Uno de los campos importantes de aplicación del PDI para el presente trabajo que a pesar de que no se utilizará en total plenitud pero ayudará al entendimiento del método usado en este trabajo, es el OCR, Reconocimiento Óptico de Caracteres, que tiene como objetivo reconocer o identificar un caracter partiendo de una imagen digitalizada el cual se representa como un conjunto de pixeles. Para realizar correctamente el proceso de OCR es necesario

analizar la imagen pixel por pixel, para esto existen cuatro pasos fundamentales (Luis Miralles Pechuán, 2015):

- Binarización: Este proceso convierte todos los pixeles de una imagen, que pueden tener varios valores, a dos valores que son 1 o 0, tomando en cuenta un umbral. En la Figura 2 (a) se muestra un ejemplo de los valores que pueden tomar los pixeles de una imagen y en la Figura 2 (b) se puede ver como se ha binarizado la imagen ya que los pixeles solamente tienen valores de 1 o 0. (El ejemplo mostrado se realizó tomando un umbral de 100, este proceso se lo explicará más adelante en este documento).

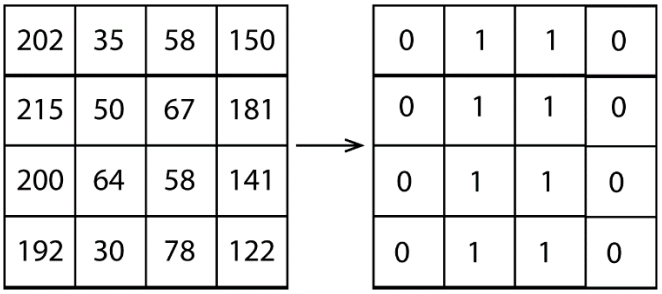
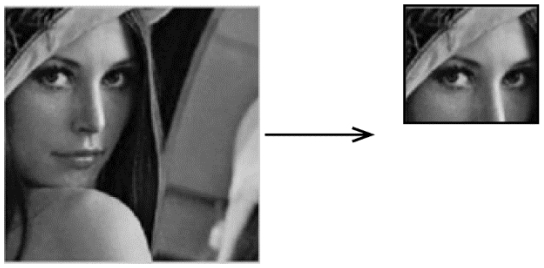


Figura 2. Binarización.

(a) Pixeles de una iamgen con diferentes valores previos a la binarización, (b) Pixeles de una imagen binarizada.

- Fragmentación: Este proceso selecciona una parte de la imagen para su análisis. En la Figura 3 se muestra un ejemplo de la fragmentación de una imagen, en la Figura 3

(a) se puede ver una imagen previa a ser fragmentada, mientras que en la Figura 3 (b) la imagen ya ha sido fragmentada para su posterior análisis.



(a) (b)

Figura 3. Fragmentación.

(a) Imagen original, (b) Imagen fragmentada.

- Adelgazamiento del componente: Mediante este proceso se logra borrar los puntos de los contornos de la imagen, hay que tener cuidado al realizar este proceso para que la imagen no pierda su forma. La Figura 4 muestra un ejemplo de este proceso.

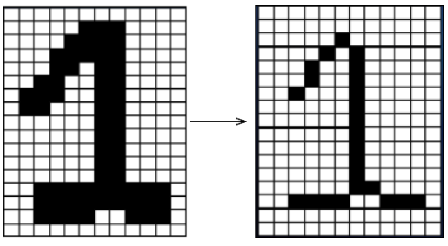


Figura 4. Adelgazamiento de componente.

- Comparación con los patrones de caracteres

Si bien en el presente trabajo de investigación no se ha usado la tecnología OCR, si se ha basado en algunos de sus pasos fundamentales, como por ejemplo el análisis pixel a pixel de la imagen para la identificación de formas o en este caso de caracteres.

Para la realización de el presente trabajo se ha usado la técnica conocida como componentes conexos la cual permite identificar rasgos de una imagen, marcándolos o etiquetándolos para de esta forma tener un conocimiento del número de rasgos encontrados y la secuencia de los mismos.

2.1.1. Componentes Conexos

Los componentes conexos asocian partes o elementos de una imagen para su posterior análisis o para cualquier tipo de operación que se pueda realizar con los mismos, los componentes conexos son “un conjunto de puntos o píxeles de las imágenes que se han agrupado a partir de cierta característica que los identifica” (Marisa R. De Giusti M. M.)

Se puede decir entonces que un componente conexo de una imagen es un conjunto de píxeles que cumple la norma que, para cada par, existe una trayectoria digital que los enlaza. Una trayectoria digital de un píxel a otro píxel es un conjunto de píxeles los cuales al pasar uno a otro el color del pixel no cambia.

Para definir lo que es un pixel, se puede ver a una imagen como un conjunto de puntos que, dependiendo del valor, indica el color del mismo. En las imágenes en escala de grises los valores son de 0 a 255 donde 0 es un punto negro y 255 un punto blanco. Como se puede observar en la Figura 5, se tiene una imagen a escala de grises donde los niveles de intensidad se representan entre 0 -255.

Figura 5. Imagen a escala de grises

El primer paso para realizar el proceso de componentes conexas de una imagen es la binarización de la misma, esto consiste en pasar de una imagen con valores de intensidad entre 0 - 255 a una imagen con valores de 0 y 1, representando de esta manera con dos niveles de gris: blanco y negro como se puede observar en la Figura 6. Para realizar este proceso se debe tomar en cuenta un umbral de binarización el cual indica que a partir de un determinado nivel de gris se toma como negro absoluto a los pixeles con menor o igual valor al umbral y en blanco total a los pixeles con valores mayores al del umbral.

Figura 6. Imagen binarizada

El proceso de binarización ayuda de gran manera al procesamiento de una imagen ya que reduce en gran cantidad los datos contenidos en la imagen.

El siguiente paso consiste en definir la vecindad de pixeles, los pixeles vecinos están condicionados al tipo de malla utilizada para representar la imagen. Por lo general y para el presente proyecto se trabaja con una malla cuadrangular la cual presenta dos posibles tipos de vecindad (Marisa R. De Giusti M. M., 2005):

- Vecindad - 4: considera los pixeles de arriba, abajo y los pixeles en los lados del pixel como vecinos (Figura 7).

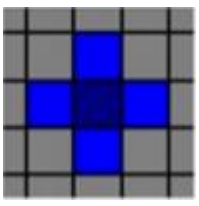


Figura 7. Vecindad - 4

- Vecindad - 8: toma como vecinos a los pixeles de 4-vecindad más los pixeles de las diagonales (Figura 8).

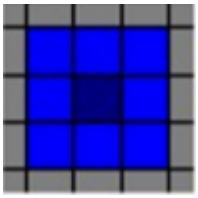


Figura 8. Vecindad - 8

A paso seguido es necesario el etiquetado de los componentes conexos ya que, si existen algunos elementos en la imagen, es indispensable diferenciar los pixeles pertenecientes a cada elemento para de esta manera trabajar en el procesamiento con los objetos deseados de la imagen. Por consiguiente este proceso consiste en establecer una misma etiqueta para todos los pixeles pertenecientes a una región conexa.

La etiqueta que se da a cada elemento tiene un nivel jerárquico que no tiene otro significado más que diferenciar los elementos de la imagen, generalmente esta etiqueta es un número.

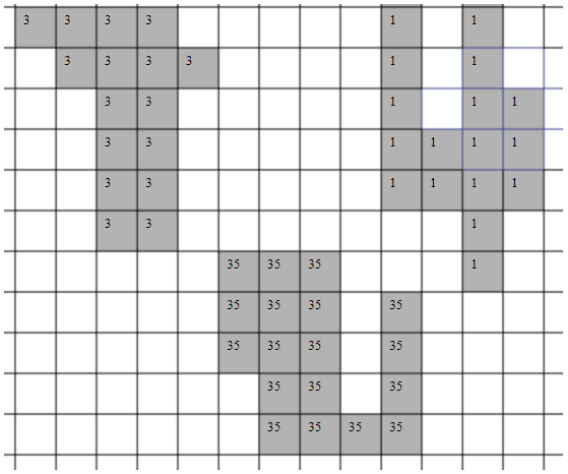


Figura 9. Etiquetado de componentes conexos

Como se puede observar en la Figura 9, la imagen ha pasado por todas las etapas anteriormente mencionadas hasta llegar al etiquetado donde se puede distinguir claramente la existencia de 3 elementos diferentes etiquetados con los numero 3, 1 y 35, esta imagen es diferente a la de las Figura 5 y Figura 6 ya que se está viendo a nivel de pixeles para poder evidenciar el etiquetado, como ya se mencionó, esta etiqueta no tiene otro significado más que identificar cada objeto o elemento de la imagen por separado.

Posteriormente a este proceso se puede realizar distintos tipos de cálculos, mediante los pixeles, para obtener diferentes características de los elementos de una imagen como por ejemplo: área, perímetro, cálculo del centroide, orientación, circularidad, envolvente convexa, obtención de *BBox* (que en adelante se utilizará como abreviativo de *bounding box*) de un caracter en caso de documentos PDF, etc. (OpenCV, 2017)

Un *BBox* puede ser definido como un cuadro delimitador el cual tiene una coordenada de inicio, ancho y alto con los que se pueden realizar las operaciones para obtener algunas de las características mencionadas en el párrafo anterior. En la Figura 10 se ejemplifica el *BBox* de un caracter, el *BBox* es el recuadro de color azul que delimita al caracter “A”.

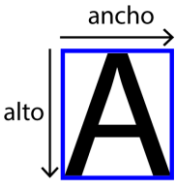


Figura 10. Ejemplo de *BBox*.

2.2. Archivos de formato PDF

Este tipo de formato en la actualidad ha tenido gran popularidad especielamente en el intercambio de información científica, por lo tanto la mayoría de documentos se encuentran en este formato y es por esto que es necesaria la investigación de documentos con este tipo formato. Este tipo de investigaciones además ayudan a la inclusión de personas con discapacidad visual ya que en la actualidad la transferencia de información en su gran mayoría se la realiza por medios digitales principalmente por archivos con formato PDF.

Un archivo en formato PDF es usado para “representar documentos de una manera independiente del software de aplicación, hardware y sistema operativo utilizado para

crearlos y del dispositivo de salida en el que se van a mostrar o imprimir" (Adobe Systems Incorporated, 2006).

El formato PDF es usado principalmente para poder manipular archivos en cualquier tipo de plataforma que se use, indistintamente de la versión o distribución del formato; esto es de mucha utilidad en la vida diaria ya que no siempre se puede contar con un programa o sistema operativo en específico para poder abrir archivos que principalmente son de lectura.

Es por esta razón que el procesamiento de este tipo de formato ha alcanzado gran popularidad aunque hay mucho campo por investigar en lo referente a la extracción de información, como por ejemplo texto, tablas, figuras y en este caso en lo que se centra este trabajo, la extracción de fórmulas matemáticas con sus respectivos caracteres.

2.2.1. Reconocimiento de fórmulas en archivos con formato PDF.

Para la detección y extracción de fórmulas matemáticas dentro de archivos con formato PDF, se ha dividido en dos diferentes tipos de fórmulas que son

- Fórmulas embebidas:

Se denota como fórmula embebida a la fórmula que se encuentra incrustada en el texto de un documento, dentro de un párrafo a línea seguida del texto, como se puede apreciar en la Figura 11 una fórmula embebida encerrada dentro de un rectángulo (Jianming Jin, 2003).

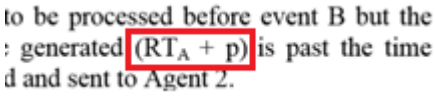
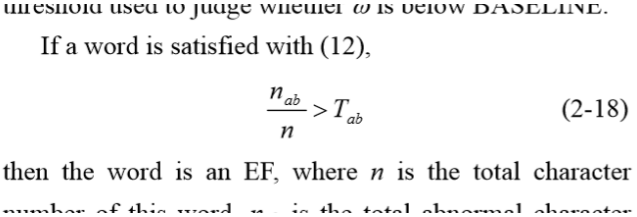

to be processed before event B but the
: generated ($RT_A + p$) is past the time
d and sent to Agent 2.

Figura 11. Ejemplo de fórmula embebida

- Fórmulas aisladas:

Se denota fórmula aislada a una fórmula que se encuentra entre dos párrafos de un texto, es decir no tiene línea seguida de texto a su lado como se ve en la Figura 12 (Jianming Jin, 2003).


threshold used to judge whether ω is below BASELINE.

If a word is satisfied with (12),

$$\frac{n_{ab}}{n} > T_{ab} \qquad (2\text{-}18)$$

then the word is an EF, where n is the total character number of this word, n_{ab} is the total abnormal character

Figura 12. Ejemplo de fórmula aislada

En el Capítulo I de este documento se ha citado varios trabajos realizados con el fin de extraer fórmulas matemáticas de archivos con formato PDF, como por ejemplo en el artículo "*Mathematical Formulas Extraction*" (Jianming Jin, 2003) se realiza la diferenciación entre fórmulas embebidas y fórmulas aisladas, basándose en la estimación de posición BASELINE/MEANLINE para las fórmulas embebidas y en el método de la ventana de

Parzen para las fórmulas aisladas, cabe mencionar que en dicho documento se realiza el procedimiento en imágenes basadas en PDFs.

El método de la ventana de Parzen toma en cuenta las dimensiones de los caracteres, ancho y alto de las líneas del documento también un promedio del ancho del total de líneas, la distancia que existe entre las líneas y el margen derecho e izquierdo del documento como se puede observar en la Figura 13.

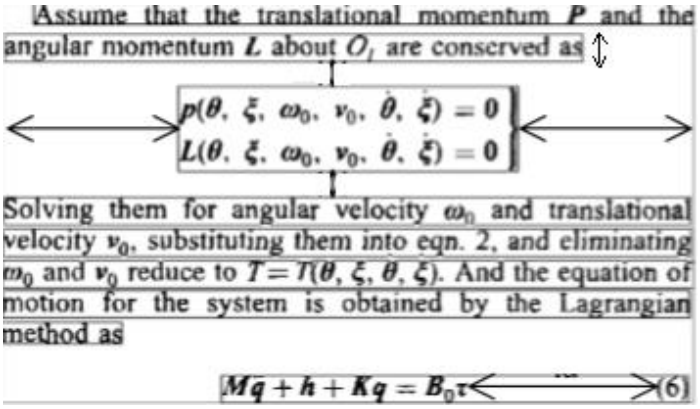


Figura 13. Características ventana de Parzen

Fuente: *(Jianming Jin, 2003)*.

Mediante todas estas consideraciones las cuales se encuentran representadas por flechas en la Figura 13 se puede estimar dónde existe una fórmula aislada en el documento.

El trabajo titulado "*Mathematical Formula Identification in PDF Documents*" (Xiaoyan Lin L. G., 2011) se centra específicamente en el procesamiento de archivos pdf donde se

menciona que los métodos que existen pueden calsificarse en tres categorías para la identificación de fórmulas dependiendo de las características usadas.

Una de las categorías identifica la región donde existe una fórmula mediante las características de los caracteres que se encuentran, por ejemplo con símbolos matemáticos, al basarse en reconocimiento de caracteres, estos métodos usan OCR por lo tanto presentan algunos inconvenientes con los elementos no reconocidos ya que existen un sin número de caracteres especiales que se usan para denotar fórmulas matemáticas, estos métodos inevitablemente presentan errores de reconocimiento.

Otra de las categorías detecta las áreas de la fórmula a través de las características de diseño, como son la altura y ancho de línea, espacio entre línea, etc. Mediante este tipo de métodos para identificar las fórmulas aisladas se sabe y es fácil notar que la altura y espacio entre líneas es mayor del promedio, generalmente se usa el método de ventanas de Parzen que ya se mencionó anteriormente (ver Figura 13).

La última categoría extrae expresiones matemáticas a través de la técnica de segmentación de imágenes, sin utilizar las características de caracter. Como se puede identificar estas tres categorías son realizadas para documentos basados en imágenes.

El método propuesto en el mismo documento (Xiaoyan Lin L. G., 2011) para la identificación de fórmulas en archivos con formato PDF consiste de cuatro pasos principales:

- Procesamiento: El fin de realizar el procesamiento es buscar que coincidan elementos con expresiones matemáticas. Una vez realizado este paso, se tendrá información en detalle como *BBox* o cuadro delimitador, línea de base, fuente y tamaño de fuente, tanto del texto como de los elementos de expresiones matemáticas.

- Detección de línea de texto: La identificación de líneas de texto beneficia la detección de otros componentes de diseño, como párrafos y columnas, además facilita la detección de fórmulas aisladas ya que se las puede identificar cuando no exista línea de texto. En la Figura 13 se puede observar un ejemplo de la línea de texto.

- Análisis de característica: En este paso se analiza según el tipo de fórmula ya sea aislada o embebida y dependiendo del caso se utiliza métodos específicos para la identificación.

- Detección del área de la fórmula: Finalmente se extrae el área que contiene la fórmula identificada ya sea aislada o embebida.

Este método concluyó con muy buenos resultados, introduciendo así un método orientado a la detección de fórmulas matemáticas en archivos con formato PDF y ya no en archivos basados en imágenes.

Como ya se ha mencionado anteriormente uno de los problemas con los que se encuentran los desarrolladores de métodos de detección y reconocimiento de fórmulas, es evaluar su trabajo y compararlo frente a otros de una forma equitativa ya que no existen bases de datos válidas y de uso libre disponibles en la literatura. Este trabajo se centra en la creación de una base de datos apoyándose en la existente (Xiaoyan Lin L. G., 2012) teniendo en cuenta los errores que se encuentren y corrigiéndolos para su posterior prueba mediante el desarrollo de una herramienta automática que permita evaluar no solo la correcta detección de fórmulas matemáticas sino también la detección e identificación de los caracteres dentro de cada fórmula, esto con la finalidad de que los desarrolladores de algoritmos de reconocimiento de fórmulas matemáticas puedan probar y compara sus trabajos con otros para de esta fórma tener un criterio equitativo del desempeño de cada algoritmo.

Uno de los posibles usos futuros de este trabajo es ayudar a desarrollar herramientas de detección de fórmulas matemáticas para personas no videntes y de esta fórma contribuir a la inclusión social y educativa de personas con discapacidad.

2.3. Discapacidad

Según la Organización Mundial de Salud OMS, discapacidad se define como “un término general que abarca las deficiencias, las limitaciones de la actividad y las restricciones de la participación. Las deficiencias son problemas que afectan a una estructura o función corporal; las limitaciones de la actividad son dificultades para ejecutar acciones o tareas, y las

restricciones de la participación son problemas para participar en situaciones vitales.” (OMS, 2017). Según la OMS una discapacidad envuelve todo un concepto de deficiencias y limitaciones indistintamente de las funciones de órganos o sentidos a las que estas afecten, y de esta manera impidiendo un desarrollo normal en situaciones de la vida cotidiana.

En la ley Orgánica de Discapacidades – LOD, en su Título II, capítulo primero, sección primera de los sujetos, en su Art.6 define a persona con discapacidad, “a toda aquella que, como consecuencia de una o más deficiencias físicas, mentales, intelectuales o sensoriales, con independencia de la causa que la hubiera originado, ve restringida permanentemente su capacidad biológica, sicológica y asociativa para ejercer una o más actividades esenciales de la vida diaria” (Asamblea Nacional del Ecuador, 2012). A diferencia del concepto dado por la OMS, la LOD especifica dos características importantes: que la afectación debe ser permanente y que no importa la causa por la que dicha afectación se haya originado.

2.3.1. Discapacidad Visual

Siguiendo las definiciones anteriormente mencionadas sobre discapacidad se puede establecer que, la discapacidad visual es la deficiencia funcional de los órganos que componen el sentido de la vista, limitando de esta forma el desenvolvimiento normal de los individuos en la sociedad, dificultando actividades fundamentales para el desarrollo normal como el aprendizaje, es por esto que es necesario la investigación e implementación de

herramientas especiales que ayuden al normal desenvolvimiento de personas con este tipo de discapacidad.

Según datos de la OMS se estima que a nivel mundial existen alrededor de 285 millones de personas con discapacidad visual, de estas cifras se sabe que al menos el 80% de las personas con ceguera son mayores de 50 años (OMS, 2017), de lo que se puede concluir que la ceguera afecta principalmente a personas de edad avanzada. Por lo tanto al tener tal cantidad de personas no videntes es importante el estudio y desarrollo de herramientas automáticas que contribuyan a su inclusión social, el presente trabajo puede contribuir en futuras investigaciones de desarrolladores de aplicaciones para la deteccion de fórmulas matemáticas para personas no videntes.

CAPITULO 3

Desarrollo

En el presente campítulo se describe la base de datos utilizada inicialmente la misma que será procesada para extraer la información necesaria y posteriormente realizar correcciones mediante un software desarrollado en este trabajo. Además se explica los tipos de correcciones realizadas para finalmente proceder a consolidar toda la información dentro de una nueva base de datos.

3.1. Descripción de la base de datos

Para este trabajo se utilizará la base de datos "marmot_data" que se la puede encontrar de forma gratuita para fines académicos[1]. Contiene información de 400 páginas de documentos en formato PDF, de los cuales 194 archivos son originados en forma digital. En total dentro de todos los documentos se encuentran 1575 fórmulas aisladas y 7907 fórmulas embebidas. La distribución de las versiones del formato PDF de los archivos originados digitalmente se puede ver en la Tabla 2 donde se evidencia que se encuentran documentos desde la versión 1.1 hasta la versión 1.6, de igual manera en la Tabla 3 se detalla los distintos productores de

[1] http://www.founderrd.com/marmot_data.htm

PDF que han generado los archivos Tener en cuenta estos detalles es importante ya que el análisis de símbolos matemáticos puede variar entre productores y versiones de PDF.

Tabla 2.

Distribución de versiones de PDF

Versión de PDF	1.1	1.2	1.3	1.4	1.5	1.6	Total
Número de archivos	1	64	66	64	1	8	194

Fuente: (Xiaoyan Lin L. G., 2012)

Tabla 3.

Distribución de Productores de PDF

Productores de PDF	Número de archivos
AFPL Ghostscript	15
Acrobat Distiller	66
Acrobat PDFWriter	10
ESP Ghostscript	23
GNU Ghostscript	10
MiKTeX pdfTeX	9
Dvipdfm	18
Dvips	4
PdfTeX	27
Others	12
Total	194

Fuente: (Xiaoyan Lin L. G., 2012)

Dentro de la base de datos cada documento PDF consta con su respectiva imagen que se han generado a 500 ppp (pixels por pulgada), estas imágenes también serán utilizadas en la realización de este proyecto, de igualmanera cada documento PDF también consta con su respectivo archivo xml dentro de deuna carpeta llamada *ground truth* que contiene la información de las fórmulas matemáticas.

El g*round truth* el cual está formado por archivos en formato XML siglas de *Extensible Markup Language*, el mismo es un metalenguaje que permite organizar y etiquetar documentos o información, los archivos pertenecientes a la base de datos se encuentran bajo el diseño de árbol que se muestra en la Figura 14, es decir tiene jerarquía. Estos archivos se los puede visualizar mediante un navegador web, en la Figura 15 se puede observar la estructura que tiene cada archivo XML de la base de datos.

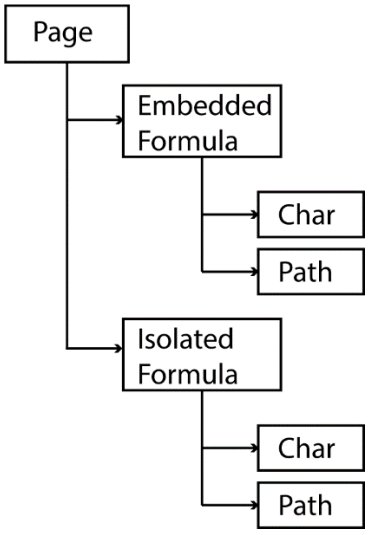


Figura 14. Diseño de árbol de archivos XML.

```
<?xml version="1.0" encoding="UTF-8"?>
- <Page PageNum="10" BBox="0000000000000000 4084500000000000 407ae00000000000 0000000000000000">
  - <EmbeddedFormula BBox="4073ddfe47beb07f 408201d91f7b9b57 4074a64c2fd40322 4081c66f37d3abe4">
      <Char BBox="407454af19ae26ef 4081d8b9ccaa681e 40747b198a9af915 4081c66f37d3abe4" Text="e" FSize="4013ECD9E83E425B"/>
      <Char BBox="4073ddfe47beb07f 4081f60d6b482284 40742912499e8490 4081d2d31d38caf8" Text="v" FSize="4023ECD9E83E425B"/>
      <Char BBox="4073e9e68c86b693 408201d91f7b9b57 40742371cea46854 4081ff609e0ff5ed" Text="" FSize="4023ECD9E83E425B"/>
      <Char BBox="40742a5ce4720500 4081d87c96d9cbb9 407744a7b76e96b69 4081c6df6fd21ff3" Text="r" FSize="4013ECD9E83E425B"/>
      <Char BBox="407485c79900ed65 4081d8b9ccaa681e 4074a64c2fd40322 4081c66f37d3abe4" Text="s" FSize="4013ECD9E83E425B"/>
    </EmbeddedFormula>
  - <EmbeddedFormula BBox="406034cbab6f59ff 407a42c0b673c4f3 4060dd7146339603 4079fc4c1a5515db">
      <Char BBox="406034cbab6f59ff 407a42c0b673c4f3 4060dd7146339603 4079fc4c1a5515db" Text="σ" FSize="4023ECD9E83E425B"/>
    </EmbeddedFormula>
  - <EmbeddedFormula BBox="407313ba816b4f84 40814adbc268c5bc 4076c81ab32f41e1 408104158933f0c8">
      <Char BBox="40749f920e4a7091 408148a076cdbcb7 4074e2adb9009220 408111caebec4cc5" Text="0" FSize="4023ECD9E83E425B"/>
      <Char BBox="4076a4b79694d767 40814adbc268c5bc 4076c81ab32f41e1 4081138bcbe61d00" Text="l" FSize="4023ECD9E83E425B"/>
      <Char BBox="4074f6bb3824df88 40811bfe8eb1084b 407507a0bdbab61f 4081138bcbe61d00" Text="," FSize="4023ECD9E83E425B"/>
      <Char BBox="40751e3b25903268 4081496c7f851b5d 4075629d449eeb68 408111caebec4cc5" Text="7" FSize="4023ECD9E83E425B"/>
      <Char BBox="4075840a74bded0b 408136c619f5748c 4075d2a036916192 408104158933f0c8" Text="p" FSize="4023ECD9E83E425B"/>
      <Char BBox="4075dd77138268d0 408148ddac9e591c 4075ff93bbc43bd9 4081138bcbe61d00" Text="i" FSize="4023ECD9E83E425B"/>
      <Char BBox="407313ba816b4f84 408135e5a9f88c6e 4073680d4ecd6d86 408112ab5be934e2" Text="σ" FSize="4023ECD9E83E425B"/>
      <Char BBox="4076066e6d2069d7 408135e5a9f88c6e 407656c50eedae99 4081138bcbe61d00" Text="x" FSize="4023ECD9E83E425B"/>
      <Char BBox="4073ced589e51ba3 408130cbd8962eab 407438d611294822 40811e25730687da" Text="=" FSize="4023ECD9E83E425B"/>
      <Char BBox="40765d2554a8a3cd 408137408596ad56 40769ad590e641bd 408112ab5be934e2" Text="e" FSize="4023ECD9E83E425B"/>
    </EmbeddedFormula>
  - <IsolatedFormula BBox="4063713f6a18779b 408038f1a9fbe76d 4078498c596a2993 407f0046f445b79a">
      <Char BBox="40778a442d816d7d 407fdb4b82b55186 4077bede6cc7d469 407f71222ce61218" Text="1" FSize="4023ECD9E83E425B"/>
      <Char BBox="406885bea86711de 407fb5d417e4f913 40692e64432b4de2 407f6f5f7bc649fb" Text="σ" FSize="4023ECD9E83E425B"/>
      <Char BBox="40700c7d5eeb47e2 407f70a7c144d94e 4070564aec728482 407f03c8b4395810" Text="I" FSize="4023ECD9E83E425B"/>
      <Char BBox="40673b0cd9c2a989 407faba075203d8c 40680f0de84b0289 407f8653aa00efe9" Text="=" FSize="4023ECD9E83E425B"/>
      <Char BBox="406acb905ac116b6 407fdb49b18f59a4 406b34c4d94de48f 407f71205bc01a36" Text="1" FSize="4023ECD9E83E425B"/>
      <Char BBox="4063713f6a18779b 407fdd87c7241661 40645956211421ca 407f712325b9cdee" Text="E" FSize="4023ECD9E83E425B"/>
      <Char BBox="4070a99aaef2c732 408024ea9ba790d3 4070f3683c7a03d2 407fdcf62a43a068" Text="I" FSize="4023ECD9E83E425B"/>
      <Char BBox="4065b99e19256af4 407f7cec05773846 40660672faff0f40 407f5856dbc9bfd2" Text="e" FSize="4013ECD9E83E425B"/>
      <Char BBox="406bafe407510226 407f9c2a326e1155 406c72adf32d5eb3 407f95c9ecb31c21" Text="−" FSize="4023ECD9E83E425B"/>
      <Char BBox="407824912b6101cc 407fe8af485787a6 4078498c596a2993 407f494879159593" Text=")" FSize="4023ECD9E83E425B"/>
      <Char BBox="406f200ed3e31690 407f70a7c144d94e 406fd339ae8233f7 407f0046f445b79a" Text="J" FSize="4023ECD9E83E425B"/>
      <Char BBox="406f46df728151cf 40801cf2447dde52 40700d704084d567 407fcfbb332c7d36" Text="+" FSize="4023ECD9E83E425B"/>
      <Char BBox="40667f0523556f9e 407fe8ad77318fc4 4066aab2342fb275 407f4946a7ef9db1" Text="]" FSize="4023ECD9E83E425B"/>
      <Char BBox="406488749f43fd85 407fe8b0412b437c 4064b421b01e405c 407f494971e95169" Text="[" FSize="4023ECD9E83E425B"/>
      <Char BBox="4077d279a999eccf 407fdb4b82b55186 407814a07d0d9cca 407f6da06cf271a2" Text="6" FSize="4023ECD9E83E425B"/>
      <Char BBox="40774dda659c66fe 407fe8af485787a6 407772d593a58ec5 407f494879159593" Text="(" FSize="4023ECD9E83E425B"/>
      <Char BBox="406d05a5d40f8c39 4080278beb9e492b 406de058c5690049 407fdcf62a43a068" Text="A" FSize="4023ECD9E83E425B"/>
      <Char BBox="406e355f1f957196 408024ea9ba790d3 406ee889fa348efd 407fd9746a4ffff2" Text="J" FSize="4023ECD9E83E425B"/>
      <Char BBox="4064cc4c00137628 407fb797c1d87d06 4065627403d31e4a 407f712325b9cdee" Text="v" FSize="4023ECD9E83E425B"/>
      <Char BBox="407040a3663695be 4080238fc0096feb 407082ca39aa45b9 407fd9746a4ffff2" Text="3" FSize="4023ECD9E83E425B"/>
      <Char BBox="4064e38178c63a62 407fcf2f2a3f6eac 40655697fd019de5 407fca3e276823d7" Text="" FSize="4023ECD9E83E425B"/>
      <Char BBox="406564f9aead2716 407f7c7199d5ff7c 4065a536d39bf3e7 407f59374bc6a7f0" Text="r" FSize="4013ECD9E83E425B"/>
      <Char BBox="406e44d1244a6224 407f6df20a08977e 406ec404f9cf6457 407f03c8b4395810" Text="2" FSize="4023ECD9E83E425B"/>
      <Char BBox="40661bcf17caf7e1 407f7cec05773846 40665cd845712359 407f5856dbc9bfd2" Text="s" FSize="4013ECD9E83E425B"/>
      <Path BBox="406cfa7ef9db22d0 407f98f9db22d0e6 4070f6c49ba5e354 407f98f9db22d0e6"/>
      <Path BBox="406aaf3333333334 408038f1a9fbe76d 407109e353f7ced9 408038f1a9fbe76d"/>
    </IsolatedFormula>
</Page>
```


Figura 15. Estructura de la base de datos, archivo: 10.1.1.1.2022_10.xml

A continuación se procederá a describir de forma general la estructura de los archivos XML. Se tiene un elemento raíz con la etiqueta “Page” el cual contiene la información de: el número de página del archivo con el nombre de atributo “*PageNum*” y el tamaño de la hoja con el nombre de atributo “*BBox*”, este atributo hace referencia a las coordenadas de las dimensiones totales de la hoja en el siguiente orden: coordenada en X inicial (Xi), ancho (Xf), alto (Yf) y coordenada en Y (Yi), estas coordenadas se encuentran dadas en la

representación hexadecimal de un número en formato IEEE-754 Floating-Point. En la Figura 16 se puede observar cómo se encuentran dispuestas las coordenadas.

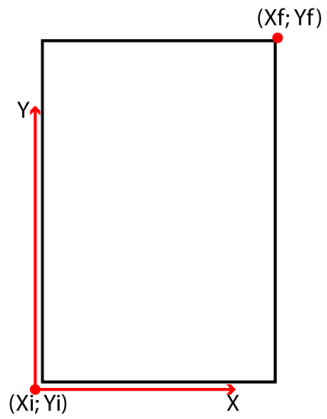


Figura 16. Descripción de coordenadas.

Como se puede observar en la Figura 15 el elemento raíz tiene etiquetas anidadas que identifican el tipo de fórmula ya sea embebida con la etiqueta "*EmbeddedFormula*" o aislada con la etiqueta "*IsolatedFormula*", cada una de estas etiquetas tiene un solo atributo con el nombre "*BBox*", que hace referencia a *bounding box* es decir el cuadro delimitador que contiene a la fórmula, el cual contiene las coordenadas de la fórmula dentro de la hoja, de igual manera este atributo se encuentra en la representación hexadecimal de un número en formato IEEE-754 Floating-Point. En la Figura 17 mediante un ejemplo se describe como se encuentran las coordenadas de las fórmulas ya sean embebidas o aisladas, la fórmula se encuentra dentro de un recuadro de color azul el cual hace referencia a su respectivo *bounding box.*

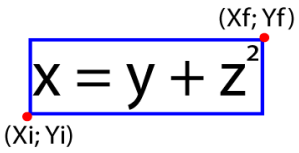


Figura 17. Descripción de coordenadas en fórmulas.

Dentro de estas dos etiquetas se pueden encontrar dos tipos de etiquetas anidadas con el nombre de "*Char*" (ver Figura 18) o "*Path*" (ver Figura 19). La etiqueta "*Char*" contiene información de los caracteres encontrados dentro de cada fórmula, con los nombres de atributos:

```
<Char BBox="407454af19ae26ef 4081d8b9ccaa681e 40747b198a9af915 4081c66f37d3abe4" Text="e" FSize="4013ECD9E83E425B"/>
<Char BBox="4073ddfe47beb07f 4081f60d6b482284 40742912499e8490 4081d2d31d38caf8" Text="v" FSize="4023ECD9E83E425B"/>
<Char BBox="4073e9e68c86b693 408201d91f7b9b57 40742371cea46854 4081ff609e0ff5ed" Text="¯" FSize="4023ECD9E83E425B"/>
```

Figura 18. Etiquetas *"Char"*.

```
<Path BBox="406cfa7ef9db22d0 407f98f9db22d0e6 4070f6c49ba5e354 407f98f9db22d0e6"/>
<Path BBox="406aaf3333333334 408038f1a9fbe76d 407109e353f7ced9 408038f1a9fbe76d"/>
```

Figura 19. Etiquetas *"Path"*.

- "*BBox*": el cual tiene una descripción idéntica al de las etiquetas anteriores.
- "*Text*": contiene el atributo del tipo de caracter en esa posición.
- "*FSize*": que contiene el atributo del tamaño de fuente el cual se encuentra en la representación hexadecimal de un número en formato IEEE-754 Floating-Point.

La etiqueta "*Path*" contiene información de las líneas horizontales que se han encontrado dentro de la fórmula a la que pertenece como por ejemplo una barra de una fracción, solamente con el atributo de nombre "*BBox*" el cual contiene el atributo de la coordenada del elemento dentro de la hoja.

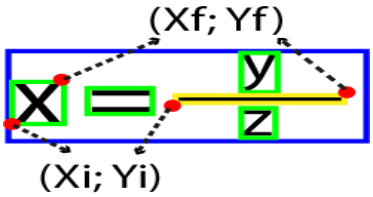


Figura 20. Descripción de coordenadas en elementos dentro de una fórmula

En la Figura 20 mediante un ejemplo se muestra la disposición de las coordenadas en los elementos dentro de una fórmula, tanto para caracteres (etiqueta *Char*) que se encuentran dentro de un recuadro de color verde el cual hace referencia a su respectivo *bounding box* y para líneas (etiqueta *Path*) se encuentran dentro de un recuadro de color amarillo el cual hace referencia a su respectivo *bounding box*, y como se explicó anteriormente el recuadro de color azul hace referencia al *bounding box* de la fórmula.

3.2. Extracción de información de la base de datos

Las especificaciones del computador con el cual se han desarrollado todos los programas del presente trabajo y software necesarios se describen en la Tabla 4.

Tabla 4.

Especificaciones del computador usado

Hardware	
RAM	4 GB
Tipo de sistema	64 bits
Procesador	Intel(R) Core™ i5 2.30 GHz
Software	
Sistema Operativo	Ubuntu 14.04
Programas instalados	• Python 2.7 • Open CV • QT Designer 4

Para poder extraer la información necesaria que está contenida dentro de la base de datos, se implementó un programa escrito en lenguaje de Python, teniendo instalada la versión 2.7 ya que más adelante se utilizarán librerías que facilitan la extracción de información dentro de archivos PDF las cuales están optimizadas para la versión 2.7 de Python. El programa crea cuatro archivos en formato CSV por cada archivo XML, cada archivo CSV tendrá por separado datos de página, datos de fórmulas aisladas o embebidas, datos de elementos dentro de la formula como caracteres y datos de elementos dentro de la fórmula como líneas. Se ha decidido separar esta información en cuatro archivos para mayor facilidad al momento de realizar una lectura de información.

Se ha decidido que los archivos tengan formato CSV siglas de *Comma Separated Values*, ya que este formato permite guardar datos en una estructura de tabla, también se los puede ver como un archivo de texto con la información contenida separada por comas. En la Figura 21 se puede observar un diagrama de los 4 archivos con formato CSV que serán creados.

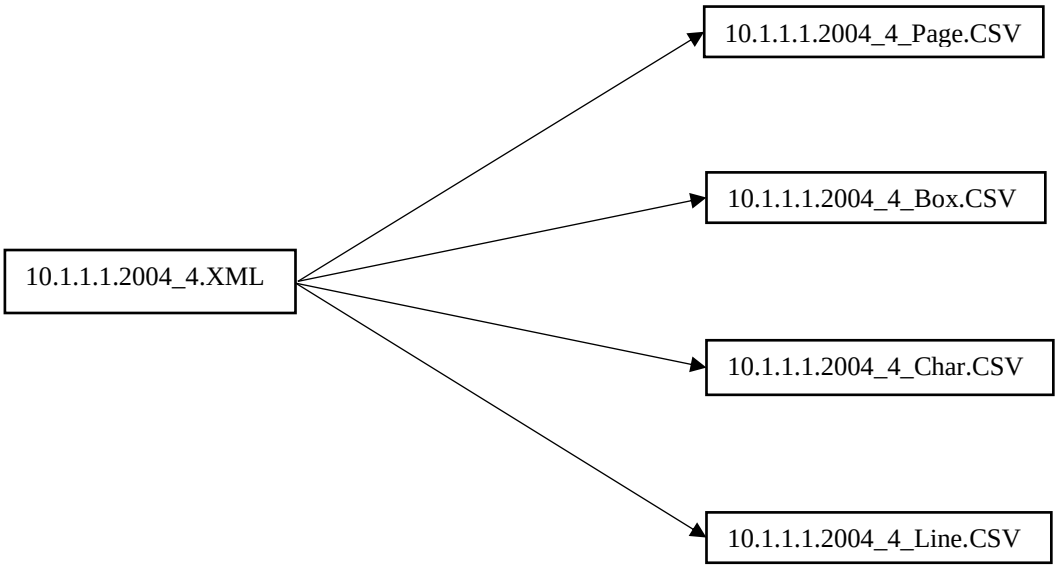


Figura 21. Diagrama de extracción de información del archivo XML

En la Figura 21, lo que se puede apreciar como "10.1.1.1.2004_4" hace referencia al nombre del archivo XML que como ya se mencionó anteriormente existen 400 archivos con este formato de los cuales se extraerá información, al final se tendrá 1600 archivos con formato CSV.

El motivo por el cual se guarda la información de un archivo XML en cuatro archivos CSV es para un posterior fácil acceso y búsqueda rápida de elementos. A continuación se describe el tipo de información que contiene cada archivo:

El archivo “10.1.1.1.2004_4_Page.CSV” contiene la información de las dimensiones de la página es decir del elemento raíz del archivo XML, el atributo “*BBox*”, pero con la diferencia de que el programa de extracción de información cambia el formato del dato de la representación hexadecimal IEEE-754 Floating-Point a Decimal Floating-Point esto con la finalidad de poder realizar cálculos posteriormente.

El archivo “10.1.1.1.2004_4_Box.CSV” contiene la información de las fórmulas en el siguiente orden: Id como identificador de la fórmula, coordenadas en formato Decimal Floating-Point y el tipo de fórmula ya sea “*Emb*” para fórmulas embebidas o “*Iso*” para fórmulas aisladas como se muestra en la Tabla 5.

Tabla 5.

Disposición de datos en archivos Box.CSV

ID	Coordenada Xi	Coordenada Yi	Coordenada Yf	Coordenada Xf	Tipo de Fórmula
1	510.2093556	415.694440625	424.247387281	524.38161975	Emb

En el archivo “10.1.1.1.2004_4_Char.CSV” se encuentra la información referente a cada caracter de cada fórmula en el orden descrito en la Tabla 6: Id de la fórmula a la que pertenece, coordenadas en formato Decimal Floating-Point, tipo de fórmula a la que pertenece el caracter ya sea “*Emb*” para fórmulas embebidas o “*Iso*” para fórmulas aisladas y finalmente el caracter codificado UTF-8, que es la codificación con la que permite trabajar Python, seguido de un ejemplo que relacionado con la Figura 22 se muestra las coordenadas y los caracteres.

Figura 22. Ejemplo para archivo Char.CSV

Tabla 6.

Disposición de datos en archivo Char.CSV

ID de fórmula	**Coordenada Xi**	**Coordenada Yi**	**Coordenada Yf**	**Coordenada Xf**	**Tipo de Fórmula**	**Caracter**
1	521.730135	416.99170	420.013385	524.38161	Emb	“c”
1	525.54562	416.32160	420.013385	528.45134	Emb	“+”
1	528.98934	416.99170	420.013385	531.23466	Emb	“x”
1	531.99936	417.15760	419.74256	534.53223	Emb	“=”
1	535.15775	415.82333	420.013385	538.23451	Emb	“2”

El archivo “10.1.1.1.2004_4_Line.CSV” contiene información que se encuentra en la etiqueta “Path” del archivo XML, de la siguiente manera: Id de la fórmula a la que pertenece, coordenadas en formato Decimal Floating-Point, tipo de fórmula a la que pertenece la línea ya sea “*Emb*” o “*Iso*”, como se puede ver esta información sigue el mismo orden que se muestra en la Tabla 5.

Para realizar el procedimiento mencionado anteriormente, se ha creado un programa llamado “leerxml.py” creado en Python 2.7 el cual se encuentra dentro del Anexo**¡Error! No e encuentra el origen de la referencia.** del presente documento, este programa realiza la extracción de información de los archivos XML y crea los cuatro archivos con formato CSV ya descritos que contendrán toda la información necesaria de las fórmulas matemáticas, mediante la utilización de la librería “*xml.etree.cElementTree*”, la cual por iteraciones permite ir avanzando entre las etiquetas y de esta forma poder extraer la información que se requiera para cada archivo CSV.

Para realizar la conversión de la representación hexadecimal de un número en formato IEEE-754 a formato decimal Floating-Point, se utilizó el módulo “`struct`” de Python el cual se usa para hacer conversiones entre valores Python y estructuras C representadas como cadenas *“strings”* de Python, también se puede utilizar en el manejo de datos binarios almacenados en archivos. De este módulo se utilizaron las funciones “*pack*” y “*unpack*” como se muestra a continuación:

```
struct.unpack('d', struct.pack('Q', int(N, 16)))
```

- N: es la representación hexadecimal de un número en formato IEEE-754, se encuentra dentro de una función *"int"* ya que al leer de la base de datos este se encuentra como una cadena de caracteres *"string"*.
- Q y d: son los formatos de desempaque y compactación respectivamente de las funciones "`pack`" y "`unpack`". En la Tabla 7 se detalla las especificaciones de estos formatos. Para el código de conversión "`d`" la representación empaquetada utiliza el formato IEEE 754 binary64.

De esta forma al ingresar la representación hexadecimal: 408033d3345069a6, se obtendrá la respectiva conversión a Decimal Floating-Point la cual es: 518.478127125

Tabla 7.

Detalle de formatos usados

Formato	Tipo en C	Tipo en Python	Tamaño estándar
d	double	float	8
Q	unsigned long long	integer	8

Fuente: (Python, 2017)

El objetivo de realizar este trabajo de investigación es la creación de una base de datos para poder usarla en herramientas que evalúen el desempeño de algoritmos de reconocimiento de fórmulas en archivos PDF, tanto de reconocimiento de posición de

fórmulas como de los elementos que estas contienen, además de reconocer el tipo de caracter o símbolo que sea dicho elemento.

La base de datos descrita en esta sección presenta múltiples errores de reconocimiento, principalmente de los elementos que contiene una fórmula. Los tipos de errores encontrados son:

- Elementos no codificados.
- Elementos con tipo de carácter o símbolo erróneos.
- Elementos codificados de forma separada, pero pertenecen a un solo elemento.

Cada uno de estos tipos de errores será descrito de forma ampliada más adelante en la sección 3.3 donde se procederá a realizar las correcciones pertinentes a la base de datos.

La metodología que se usará para realizar la corrección de la base de datos será de la siguiente manera. En primer lugar se extraerá toda la información que contiene cada archivo XML que se encuentra dentro de la base de datos, para así poder conocer las posiciones de cada fórmula dentro de un documento y proceder a extraerlas. Una vez realizada la extracción de la fórmula se procederá a realizar una verificación visual de la correcta detección de cada uno de los elementos que se encuentran dentro de la fórmula, en caso de que existan elementos que no hayan sido detectados, se realizará de forma manual la detección para posteriormente ingresarlos a la base de datos. De igual manera se realizará una verificación

visual de que el tipo de carácter o símbolo de cada elemento que se encuentra en la base de datos sea el correcto, en caso de no ser correcto se realizará la corrección necesaria.

Todas las correcciones que se realicen a la base de datos son con el fin de obtener una base de datos fiable para que a futuro pueda ser usada en herramientas que permitan realizar una evaluación de algoritmos de reconocimiento de fórmulas en archivos PDF, que principalmente se enfoquen en la detección de elementos dentro de las fórmulas.

3.3. Corrección de Datos

Para realizar la corrección de la información obtenida de la base de datos, se desarrolló un software el cual por medio de una interfaz gráfica permite distintas funcionalidades de uso tanto como para la verificación, corrección y en caso de ser necesario añadir nueva información.

Para la implementación de la interfaz gráfica se utilizó el software QTDesigner 4 que es la herramienta de Qt[2] el cual viene incluido al instalar PyQt4[3] que es un software de Qt específico para Python, QTDesigner permite diseñar y construir interfaces gráficas de usuario

[2] https://www.qt.io/
[3] https://sourceforge.net/projects/pyqt/files/PyQt4/PyQt-4.11.4/

mediante un entorno gráfico. A la vez que esta interfaz gráfica posteriormente deberá ser ejecutada mediante programación en Python que se mencionará más a delante.

Con la finalidad de realizar las correcciones de forma más rápida y para facilitar el uso de la herramienta de corrección en un sistema operativo Microsoft Windows se creará una aplicación con extensión .exe de esta forma la herramienta que se creará podrá ser ejecutada en cualquier máquina que tenga un sistema operativo de Microsoft Windows.

3.3.1. Interfaz Gráfica

Para realizar las correcciones pertinentes, es necesario visualizar cada fórmula que contienen los documentos PDF de forma gráfica, para posteriormente realizar la inspección visual, por lo tanto también es necesario verificar los elementos que contiene la fórmula y los tipos caracteres detectados que se encuentran almacenados en la base de datos. Para ello será necesario realizar un barrido entre todas las fórmulas que existan dentro de un documento y a la vez un barrido dentro de los elementos que se encuentran dentro de cada fórmula.

Otra opción necesaria para realizar las correcciones será codificar nuevos elementos que no estén detectados en la base de datos, así como también poder unir elementos que se encuentren o no dentro de la base de datos.

Se requiere cubrir todas las necesidades explicadas mediante una sola interfaz gráfica, para el diseño de dicha interfaz, se utilizará la herramienta QTDesigner 4. En la Figura 23 se muestra un esquema de cómo sería el diseño final de la interfaz, donde se puede visualizar 5 secciones. La primera sección de la figura deberá mostrar el documento miniaturizado del que se está extrayendo las fórmulas. En la segunda sección serán necesario botones que permitan realizar el barrido de fórmulas dentro del documento y el barrido de elementos dentro década fórmula. La tercera sección está destinada para visualizar el tipo de caracter que se encuentra almacenado en la base de datos para poder verificar si es el correcto o no, también se incluirá la opción para ingresar la corrección y guardarla. En la sección número cuatro existirá la opción de realizar la codificación o detección de un nuevo elemento que no se encuentre en la base de datos, de igual manera se incluirá la opción de unir elementos si fuera necesario. En la quinta sección se deberá visualizar la fórmula en la que se está realizando la inspección y también será necesario poder verificar cual es el elemento dentro de la fórmula que se está mostrando como caracter en la tercera sección.

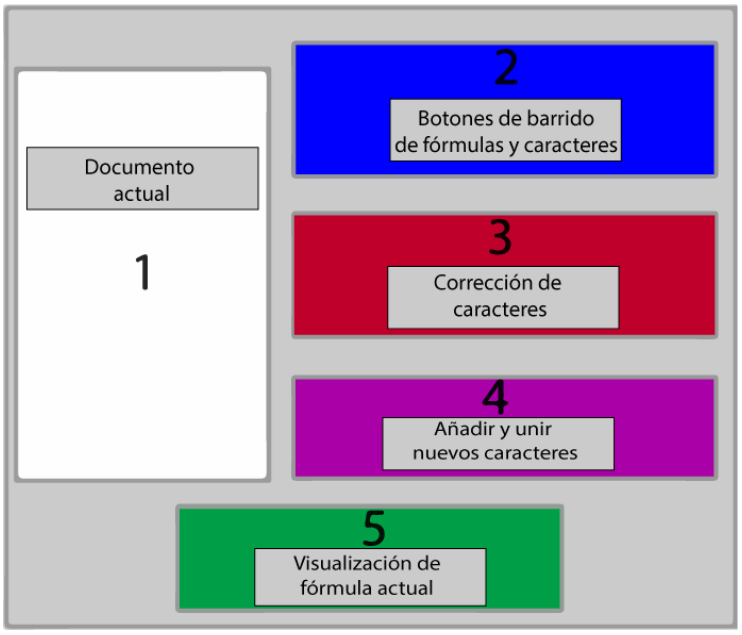


Figura 23. Diagrama de la interfaz gráfica.

Para realizar todo lo requerido, mediante la herramienta QTDesigner 4 se realizó la interfaz gráfica que se muestra en la Figura 24, esta interfaz consta con una opción desplegable de menú y cinco secciones, al igual que el diagrama mostrado en la Figura 23, las cuales se describirán en las siguientes secciones del documento. La programación usada para realizar todas las funciones de la interfaz se la puede verificar en el Anexo**¡Error! No e encuentra el origen de la referencia.** de este documento.

La interfaz fue realizada mediante un entorno gráfico que facilita la herramienta QTDesigner, tomando en cuenta diferentes opciones para su ordenamiento y facilidad de uso entre los cuales se puede mencionar: *Layouts* horizontales y verticales, separar la ventana en diferentes secciones mediante el uso de *"Combo box"*.

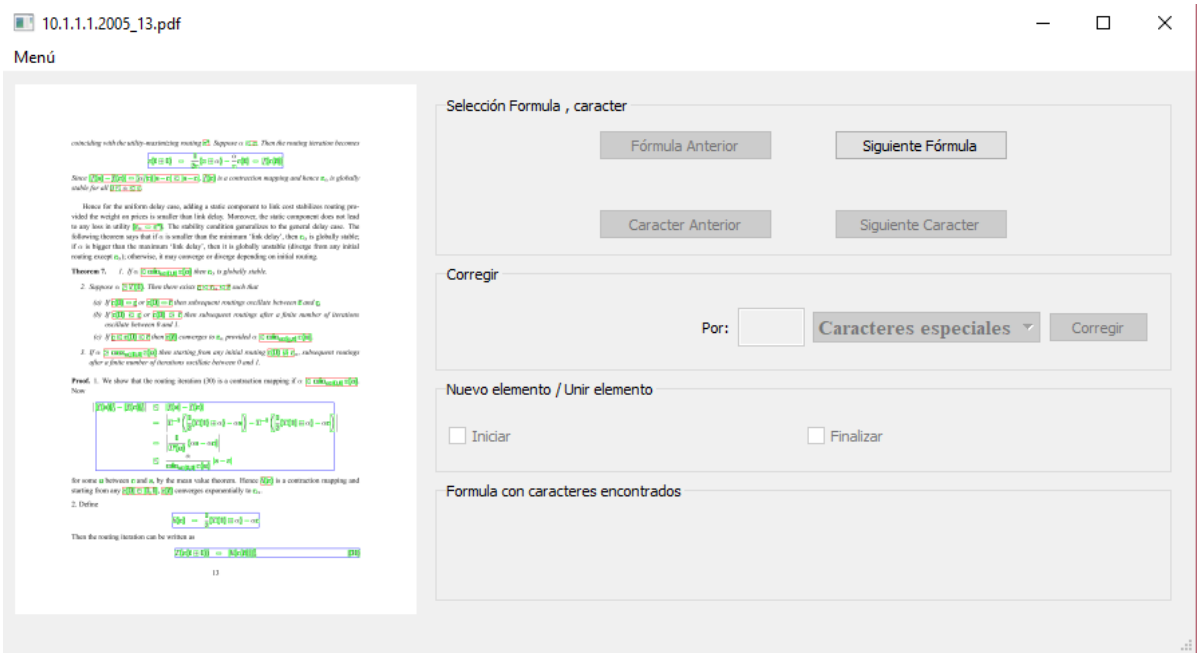


Figura 24. Interfaz gráfica

Como se observa en la Figura 25, al desplegar la pestaña de "Menú", esta presenta dos opciones básicas:

Figura 25. Opciones pestaña de Menú

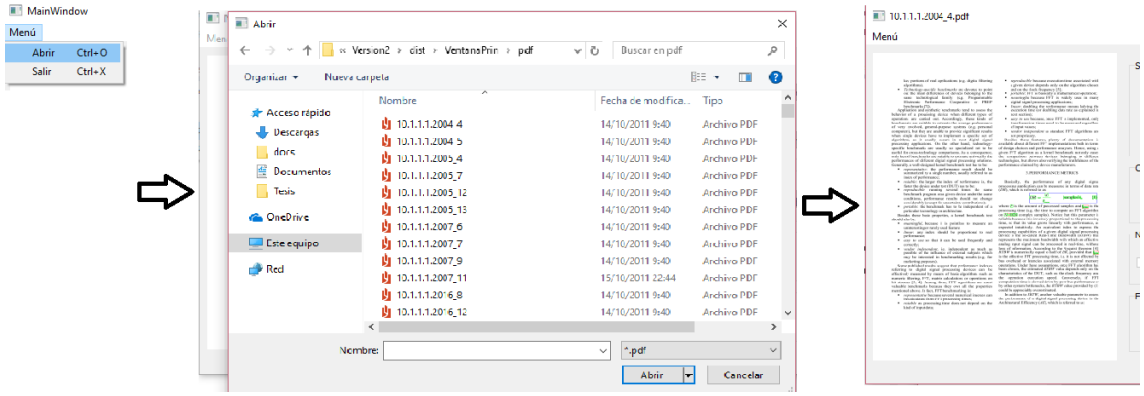
- Abrir: esta opción mediante la respectiva programación en Python, permite elegir únicamente archivos con formato PDF y que pertenezcan a la base de datos, si el archivo que se ha elegido no es uno de los archivos pertenecientes a la base de datos, mediante un cuadro de dialogo se informará que solo se pueden abrir dicho archivo, ya que si se selecciona cualquier otro archivo PDF, no se encontrará el archivos CSV (creado anteriormente, ver sección 3.2) para poder obtener información de las fórmulas.
- Salir: esta opción puede ser programada directamente desde la herramienta QTDesigner para cerrar la ventana.

En la Figura 26 se puede evidenciar como una vez elegido el archivo que se desea abrir, mediante programación se obtiene el nombre sin ninguna extensión para posteriormente abrir los archivos de la base de datos relacionados con el nombre elegido tanto la imagen con extensión TIF[4] (*Tagged Image File Format)* que se despliega en miniatura en el recuadro blanco de la parte izquierda de la ventana (ver Figura 24) como los archivos CSV creados (ver Figura 21).

Una vez elegido el archivo se utilizará la función "`leercsv`" del programa "imgrect.py", que permite leer las coordenadas de cada fórmula de la base de datos, y se procede a dibujar los *BBox* de cada fórmula del documento para que la imagen que se despliegue tenga marcados los *BBox* de las fórmulas de color azul para fórmulas aisladas (ver Figura 27 (a)) y

[4] Es un formato de archivo etiquetado, usado para realizar procesamiento gráfico de alta gama.

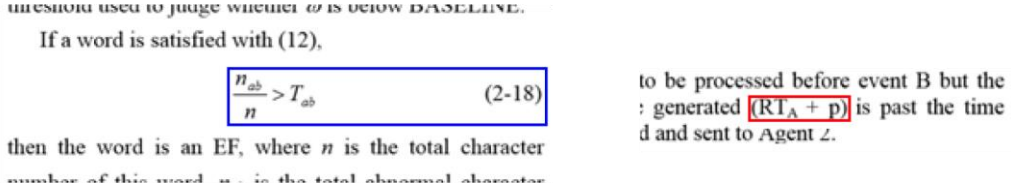
de color rojo a las fórmulas embebidas (ver Figura 27 (b)), además se habilitará el botón de Siguiente fórmula de la sección de “Selección fórmula, caracter”.



(a) (b) (c)

Figura 26. Abrir documento.

(a) Se selecciona la opción abrir del menú, (b) Cuadro de dialogo para seleccionar archivo, (c) Imagen del archivo PDF seleccionado.



If a word is satisfied with (12),

$$\frac{n_{ab}}{n} > T_{ab} \qquad (2\text{-}18)$$

then the word is an EF, where n is the total character

to be processed before event B but the
: generated $(RT_A + p)$ is past the time
d and sent to Agent $\mathcal{Z}$.

(a) (b)

Figura 27. *BBox* de los diferentes tipos de fórmulas.

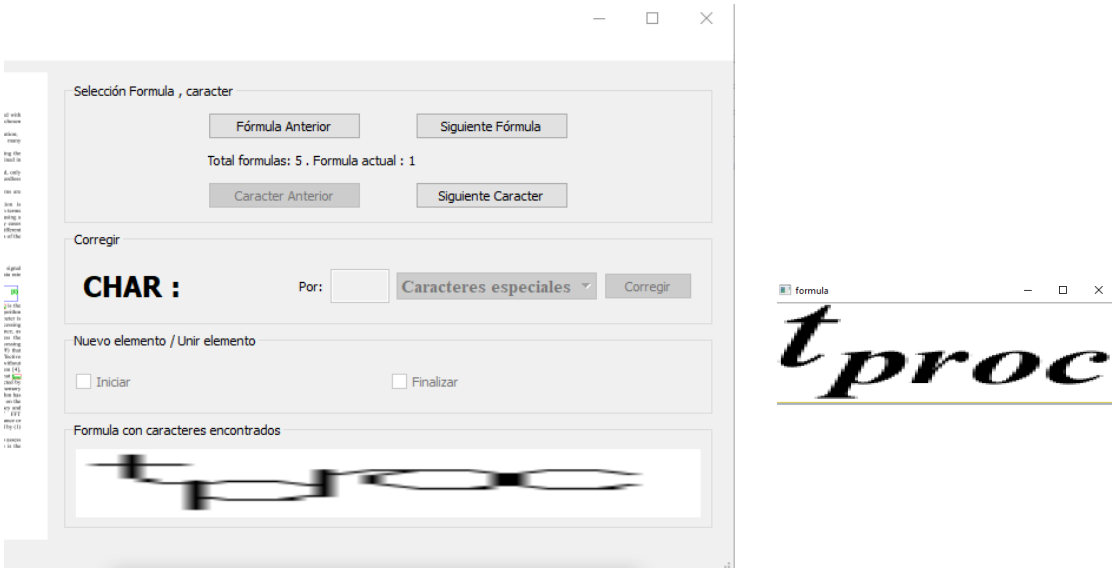
(a)*BBox* para fórmulas aisladas, (b) *BBox* para fórmulas embebidas.

A continuación se detallará el funcionamiento de cada una de las cuatro secciones restantes de la interfaz gráfica.

3.3.1.1. Sección: Selección Fórmula, caracter

En esta sección se muestra los botones que permitirán seleccionar una fórmula y desplazarse entre sus caracteres, en total existen cuatro botones (ver Figura 24) y un texto informativo. Como ya se mencionó el primer botón que se habilitará una vez seleccionado el archivo deseado es el de Siguiente fórmula, la funcionalidad de cada elemento de esta sección es la siguiente:

- Siguiente Fórmula: Al accionar este botón
    - Se iniciará el barrido en las fórmulas dentro del archivo, como se muestra en la Figura 28 (a), existe un indicador que se encuentra bajo este botón que indica cuantas fórmulas existen y en cuál se encuentra actualmente.
    - Se habilitarán los botones de Fórmula anterior y Siguiente caracter.
    - Se abrirá una ventana la cual contiene la fórmula extraída del documento que se analizará (ver Figura 28 (b)) esto se logra mediante herramientas de Python y OpenCV siglas de *Open Source Computer Vision Library,* que es una librería desarrollada para varios lenguajes de programación que proporciona herramientas para el procesamiento digital de imágenes,.



(a) (b)

Figura 28. Acción botón "Siguiente fórmula".

(a) Interfaz gráfica, (b) Ventana auxiliar que muestra la fórmula seleccionada.

- Fórmula Anterior: Tiene la función de retroceder entre las fórmulas del documento, de igual forma que el botón descrito anteriormente, la fórmula que se va a analizar se extraerá en una nueva ventana auxiliar.

- Siguiente Caracter: Este botón permite realizar un barrido dentro de los caracteres de la fórmula actual y de esta forma poder verificar su correcta codificación dentro de la base de datos. Como se puede observar en la Figura 29(a), carácter actual se muestra el *BBox* en color azul, carácter anterior se muestra en color verde (Figura 29 (b)). Esto permite un indicador de que ya se analizó el caracter y de esta manera al

final poder observar que se haya analizado todos los caracteres de la fórmula (Figura 29 (c)). Una vez que se haya finalizado el barrido por todos los caracteres que se encontraron en la base de datos, el botón se deshabilitará.

- Caracter Anterior: Este botón se habilitará una vez que se accione el botón de Siguiente caracter y cumple la misma función del botón Siguiente caracter con la diferencia de que en lugar de avanzar, regresa en los caracteres ya analizados.

- Campo de texto: Este texto aparece una vez que se ha accionado el botón de Siguiente fórmula, y tiene la función de indicar el número total de fórmulas que se han encontrado en la base de datos pertenecientes al documento seleccionado, también indica el número de la fórmula en la que se encuentra.

(a) (b) (c)

Figura 29. Barrido de elementos dentro de la fórmula.

(a) Se inicia el barrido de los elementos, (b) Se puede observar como una vez que se pasa de un elemento, el *BBox* de este cambia de color a verde, (c) Culmina el barrido de elementos.

3.3.1.2. Sección: Corregir

En esta sección se procederá a corregir la codificación de los elementos de las fórmulas en caso de que sea necesario. A continuación se muestra un ejemplo donde es necesaria la corrección de la codificación de una fórmula matemática. En la Figura 30 (a) se puede observar una fórmula matemática extraída de un documento PDF y en la Figura 30 (b) se muestra los caracteres que han sido obtenidos de la base de datos pertenecientes a la fórmula, como se puede verificar el exponente de la variable “y” que se encuentra encerrado con un rectángulo de color rojo, está mal codificado ya que en lugar de ser “2”, en la base de datos se encuentra como “z”, por lo tanto es necesaria la corrección que consiste en reemplazarel caracter “z” por el caracter “2”.

(a) (b)

Figura 30. Explicación de corregir.

(a) Fórmula matemática extraida de un documento PDF, (b) Fórmula con caracteres codificados dentro de la base de datos.

Como muestra la Figura 31, esta sección está formada por un texto informativo, un cuadro para ingresar texto (*Line edit*), una lista de caracteres especiales y un botón. Esta sección permitirá realizar correcciones de los caracteres codificados en la base de datos.

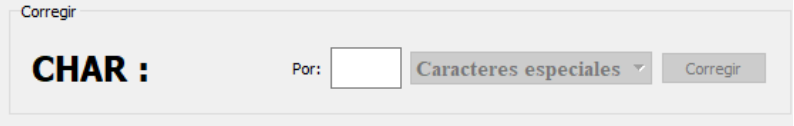


Figura 31. Sección Corregir

La función de esta sección inicia cuando se acciona el botón Siguiente caracter, como se puede ver en la Figura 32 , el caracter que se está analizando se encuentra con el *BBox* marcado de color azul, en este caso mediante el análisis visual se puede concluir que el caracter es el número 4 lo que concuerda con la información desplegada en la sección "Corregir" en el texto informativo *CHAR: "4"*, donde se muestra la información obtenida de la base de datos por lo tanto no es necesario realizar una corrección.

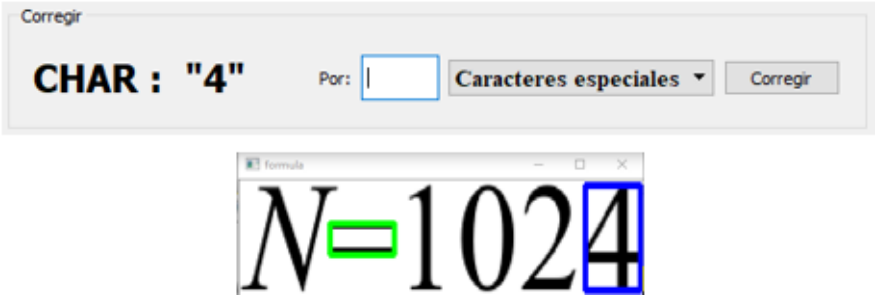


Figura 32. Ejemplo sin corrección

Cuando es necesario corregir un elemento mal codificado, como se muestra en la Figura 33. El caracter que se reconoce visualmente como una "," (coma) en la base de datos se encuentra codificado como ";" (punto y coma) como se muestra en el texto informativo *CHAR: ";"*. Para esto existen dos posibles formas de realizar la corrección:

a) Si el caracter por el cual se va a corregir se lo puede ingresar desde el teclado, se lo ingresa en un *Line edit* que se muestra en la Figura 33 con un recuadro azul.

b) En caso de que el caracter que se desea ingresar no se encuentre en el teclado, existe una lista con el nombre de “Caracteres especiales” donde al desplegarlo se enlistan símbolos matemáticos los cuales no se encuentran en el teclado y serán necesarios para la correcta codificación de los caracteres. La Tabla 8 muestra los caracteres especiales que se han enlistado.

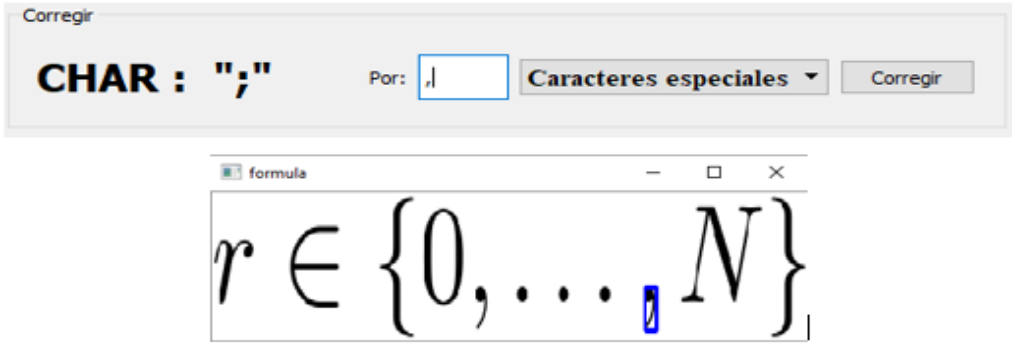


Figura 33. Ejemplo con corrección

Tabla 8.

Caracteres especiales enlistados.

#	Caracter especial	#	Caracter especial	#	Caracter especial
1	√	***27***	∉	***53***	◅
2	Π	***28***	∄	***54***	▻
3	Σ	***29***	∴	***55***	⊥
4	β	***30***	∵	***56***	≃
5	σ	***31***	∮	***57***	ψ
6	τ	***32***	∫	***58***	¢
7	∈	***33***	∇	***59***	÷
8	≤	***34***	⊣	***60***	∃
9	≥	***35***	π	***61***	←
10	λ	***36***	≈	***62***	↑
11	Δ	***37***	∩	***63***	∠
12	ξ	***38***	∅	***64***	≙
13	∧	***39***	→	***65***	≪
14	≡	***40***	↓	***66***	≫
15	~	***41***	↔	***67***	∪
16	∀	***42***	↕	***68***	⊆
17	ρ	***43***	±	***69***	⊄
18	α	***44***	η	***70***	⊂
19	θ	***45***	£	***71***	⊇
20	≠	***46***	Φ	***72***	⊢
21	Θ	***47***	ç	***73***	⊗
22	ω	***48***	γ	***74***	⊕
23	∂	***49***	δ	***75***	○
24	μ	***50***	⁒	***76***	●
25	⊃	***51***	∞		
26	⊅	***52***	φ		

- Botón Corregir: Para esta sección la función de este botón es guardar las correcciones que se realicen, en primera instancia se verifica si se ha ingresado un caracter valido para la corrección ya que no se permite el uso de tildes, se verifica también que solo se ingrese por el *Line edit* o por el listado de caracteres especiales ya que no es posible utilizar las dos opciones al mismo tiempo, en caso de suceder alguna de estas fallas al ingresar la corrección, una ventana de dialogo informará el tipo de error que se está cometiendo.

  Una vez realizadas las verificaciones, se procede a crear un archivo con formato CSV con nombre "Corregir" el cual tendrá la información necesaria para saber el nombre del archivo PDF, la fórmula y el caracter que se está corrigiendo. La Tabla 9 muestra los detalles que se almacenan en el archivo CSV.

Tabla 9.

Disposición de datos en archivo Corregir.CSV

Nombre de Documento PDF	**ID de fórmula**	**Coordenada de caracter**	**Tipo de fórmula**	**Caracter de la base de datos**	**Coordenada de la fórmula**	**Caracter por el que se va a corregir**
10.1.1.1.2005_4	4	Coordenadas Xi,Xf,Yi,Yf	Emb	";"	Coordenadas Xi,Xf,Yi,Yf	","

3.3.1.3. Sección: Nuevo elemento / Unir elemento

En esta sección mediante el uso de técnicas de procesamiento de imágenes, como la de componentes conexos descrita anteriormente en el CAPITULO 2, se procederá a agregar elementos que no se encuentran codificados dentro de una fórmula, o a unir elementos que estén codificados por separado pero en realidad deben ser uno solo. En la Figura 34 se ejemplifica el proceso que se realizará para unir elementos, como se puede ver el caracter "i" debería tener un solo *BBox* que lo contenga, pero en su lugar existen dos *BBox* (ver Figura 34 (b)) lo cual es incorrecto, por lo tanto la correcta codificación de *BBox* debería ser la que se aprecia en la Figura 34 (c).

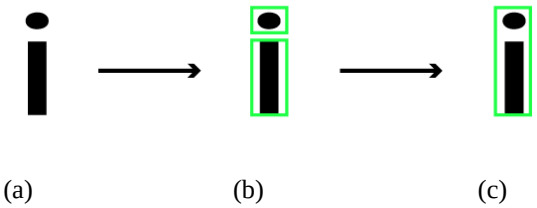


Figura 34. Explicación para unir elemento

(a) Caracter que debería tener un solo *BBox*, (b) *BBox* de caracter incorrecto por separado, (c) *BBox* de caracter correcto después de ser unido.

Por otra parte en caso de que el *BBox* del carácter no se encuentre almacenado en la base de datos será necesario codificarlo y guardarlo como un nuevo elemento.

Para realizar estos dos procedimientos, será necesario el uso de la librería OpenCV ya que mediante programación realizada en Python, se realizará el procedimiento necesario para encontrar componentes conexos en las fórmulas.

A continuación se describe el procedimiento.

En primera instancia se procede a obtener la imagen en escala de grises mediante el comando:

```
imgGray = cv2.cvtColor(imgInput, cv2.COLOR_BGR2GRAY)
```

Este comando convierte una imagen BGR a escala de grises:

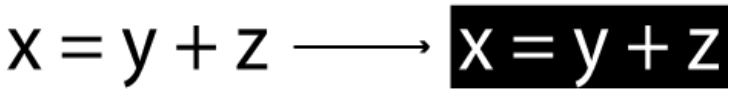
- `cv2.cvtColor`: Función de la librería *OpenCV* que permite realizar conversiones de una imagen.
- `imgInput`: Es el nombre de la imagen que se desea convertir a esacala de grises.
- `cv2.COLOR_BGR2GRAY`: Función de la ibreria *OpenCV* que mermite realizar la conversión de una imagen BGR a escala de grises.

Posteriormente se procede a realizar la binarización de la imagen mediante el comando:

```
cv2.threshold(imgGray, 125, 255, cv2.THRESH_BINARY_INV)
```

- `cv2.threshold`: Función de la librería OpenCV que permite realizar la binarización.
- `imgGray`: Nombre de la imagen que se desea binarizar.
- `125`: Valor umbral, los pixeles que tengan un valor menor o igual al umbra tomarán un valor de 0 y los pixeles que tengan un valor mayor al umbral tomarán un valor de 1.
- `255`: Valor máximo de los pixeles en la imagen, en una imagen a escala de grises los pixeles tienen un valor de 0 a 255.
- `cv2.THRESH_BINARY_INV`: Parametro de la librería OpenCV que define el tipo de binarización que se realizará.

  Se realiza una binarización invertida es decir a lo blanco se hara negro y viceversa (ver Figura 35), esto debido a que seguido se utilizará una función de OpenCV la cual requiere estas características.



(a) (b)

Figura 35. Imagen con binarización invertida

(a) Imagen a escala de grises, (b) Imagen con binarización invertida.

Una vez obtenida la imagen binarizada, se procede a encontrar los contornos que posee la imagen, los contornos se pueden explicar simplemente como una curva que une todos los

puntos continuos, teniendo el mismo color o intensidad. Esto se logra mediante el siguiente comando:

```
cv2.findContours(rectdilation.copy(),cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
```

- `cv2.findContours`: Función de la librería *OpenCV* que permite realizar la búsqueda de contornos en una imagen, para obtener una mayor precisión, es necesario usar imágenes binarizadas. El buscar contornos se puede decir que es buscar objetos blancos en un fondo negro, es por esto que en el preoceso de binarización se ustilizo la binarizacion inversa.
- `rectdilation`: Imagen en la que se buscará los contornos.
- `cv2.RETR_EXTERNAL`: Es el modo de recuperación de contornos, este modo recupera solo los contornos exteriores extermos y establece una jerarquía de -1 para todos los contornos.
- `cv2.CHAIN_APPROX_SIMPLE`: Es el método que permite usar la librería, para obtener los contornos de una forma optima para el uso de memoria ya que no almacena todos los puntos del contorno de una imagen, si no almacena solo los puntos de curvas que posee la imagen.

La Figura 36 muestra un ejemplo si se dibujara los contornos extraidos con color verde.

Figura 36. Ejemplo de dibujar un contorno.

Finalmente se realiza la identificación del *BBox* que contiene a cada elemento de la imagen o de la fórmula. Esto se logra mediante el comando:

```
[intX, intY, intW, intH] = cv2.boundingRect(Contornos)
```

- `cv2.boundingRect`: Función de la librería *OpenCV*, la cual por medio de los contornos obtiene coordenadas del *BBox* que contiene a dicho contorno.
- `[intX, intY, intW, intH]`: Las coordenadas del *BBox* obtenido, donde `intX` e `intY` son las coordenadas iniciales de la esquina superior izquierda del elemento, `intW` e `intH` es el ancho y el alto respectivamente. En la Figura 37 se puede apreciar que cada elemento de la imagen esta contenido en su *BBox* exacto el cual se encuentra representado de color verde.

Figura 37. Imagen con *BBox* de cada elemento.

Todo el proceso en el cual se realiza la técnica de componentes conexos se encuetra documentado en los Anexos, función "`conexos`".

Utilizando la explicación dada de como encontrar los *BBox* de cada elemento de una fórmula mediante componentes conexos, a continuación se describirá la sección "Nuevo elemento / Unir elemento" cuyo objetivo es agregar a la base de datos elementos que no estén codificados o unir elementos que estén codificados erróneamente por separado.

Como se puede ver en la Figura 38, en esta sección se tienen solamente dos opciones mediante dos *checkboxs* llamados "Iniciar" y "Finalizar", y se habilita una vez que se acciona el botón de Siguiente Caracter, estos *checkbox* tienen una funcionalidad exclusiva, es decir solamente se puede marcar uno a la vez.

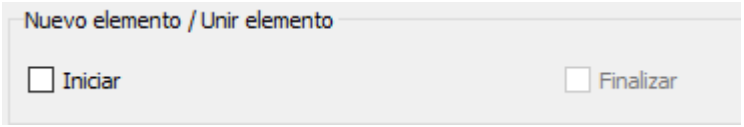


Figura 38. Sección Nuevo elemento / Unir elemento

A continuación se detalla cómo se realiza las dos funciones de esta sección:

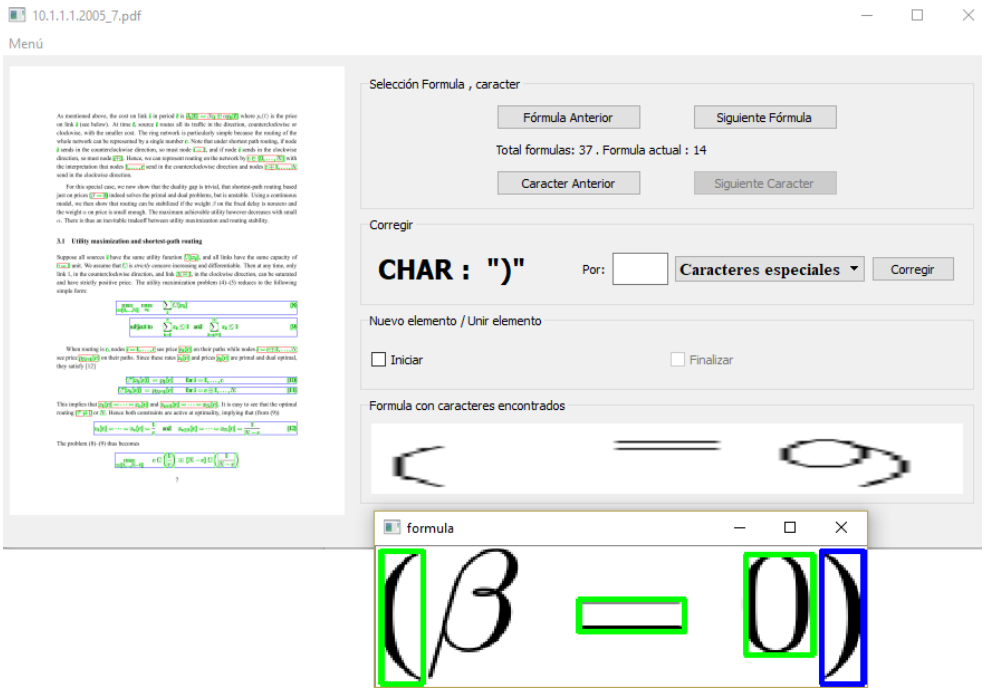


Figura 39. Ejemplo nuevo elemento

a) Nuevo elemento: En la Figura 39 se puede observar que existe un caracter que no se ha marcado (β), esto indica que este caracter no está codificado en la base de datos y por lo tanto hay que proceder a codificarlo. A continuación se detalla los pasos a seguir para realizar este procedimiento:

- Marcar el *checkbox* Iniciar.
- Dar "*click"* sobre el elemento dentro de la fórmula que desea agregar a la base de datos, este se marquará de color rojo para indicar que se está tomando como nuevo elemento. Este procedimiento se lo realiza mediante el uso de la técnica de componentes conexos explicada anteriormente.

- El programa verificará mediante un cuadro de dialogo si se ha marcado el elemento que se desea agregar como nuevo.
- En la sección de Corregir se procede a ingresar el caracter con el cual se guardará.
- Marcar el *checkbox* Finalizar y se guardará la información del elemento en el archivo “Corregir.CSV” descrito anteriormente. Con un cambio que en la descripción de “Caracter de la base de datos” (ver Tabla 9) estará como “Nuevo unido”. Esta acción también se la puede realizar mediante el botón de Corregir

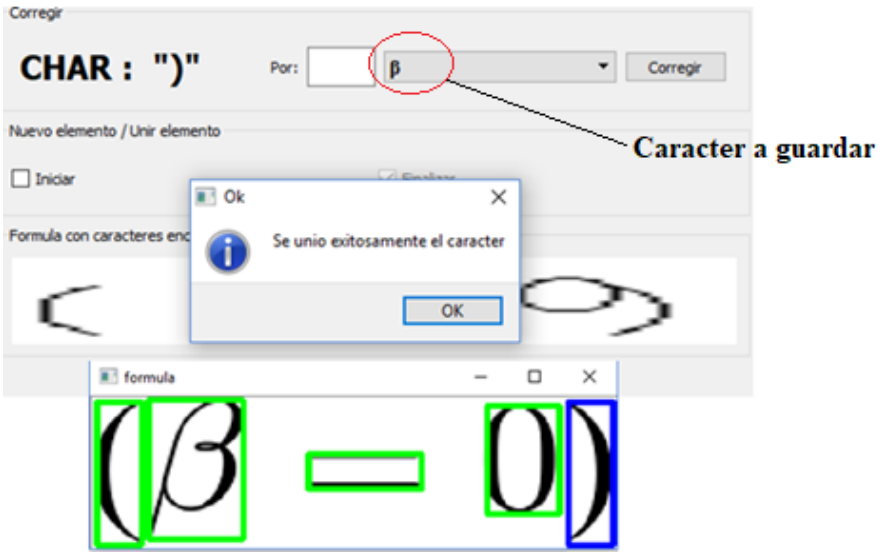


Figura 40. Elemento agregado correctamente

Una vez que se haya agregado correctamente el elemento, una ventana de información indicara que no sucedió ningún error, y también se podrá verificar visualmente ya que como se puede ver en la Figura 40, el *BBox* del elemento que se seleccionó cambiará de color rojo a color verde.

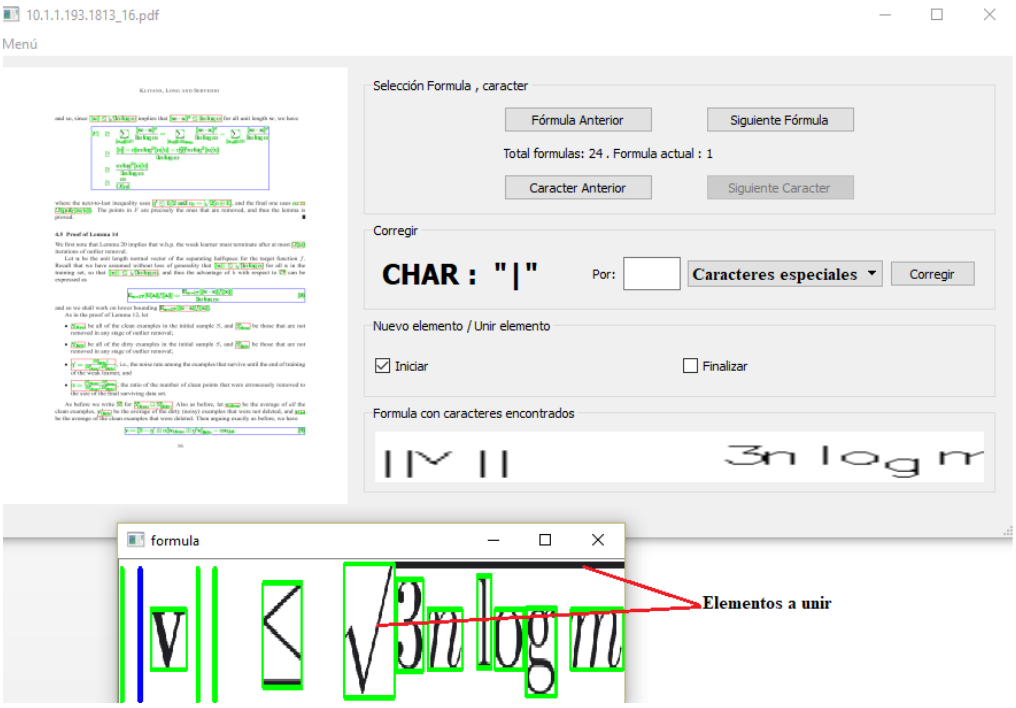
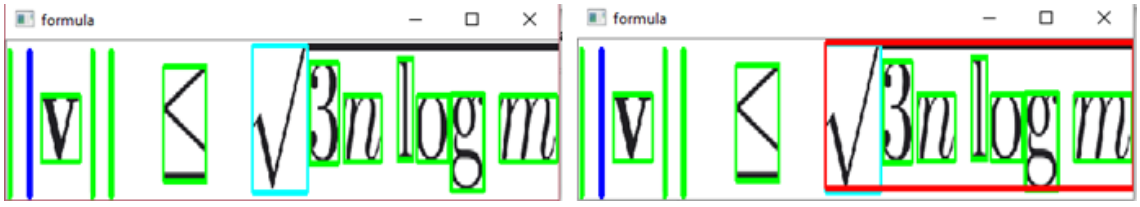


Figura 41. Ejemplo unir elemento

b) Unir elemento: En la Figura 41 se puede observar que ya se ha realizado el barrido por todos los elementos de la fórmula sin embargo quedan elementos que no están codificados en su totalidad, como en este caso el símbolo de raíz cuadrada no está totalmente codificado, por lo tanto se procede a unir los elementos necesarios. A continuación se detalla los pasos a seguir para realizar este procedimiento:

- Marcar el *checkbox* Iniciar.
- Marcar los elementos que se desea unir mediante un "*click*" sobre el cada elemento. Cuando se marca un elemento que ya se encuentra codificado, el *BBox* de este cambiará de color verde a celeste (ver Figura 42 (a)), cuando se marca un elemento que no está codificado, el *BBox* de este será de color rojo (ver Figura 42 (b)).
- En la sección de Corregir se procede a ingresar el caracter con el cual se guardará el nuevo elemento unido.
- Marca el *checkbox* de Finalizar.



(a) (b)

Figura 42. Marcar elementos a unir

Posteriormente si no existe ningún error el momento de guardar los cambios, los elementos unidos tendrán un solo *BBox* de color verde como se muestra en la Figura 43.

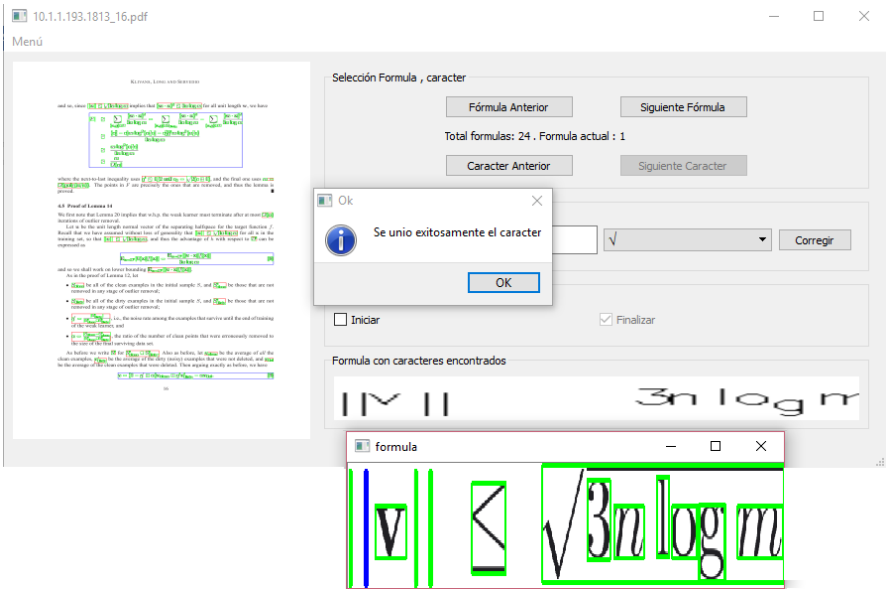


Figura 43. Elementos unidos correctamente

El nuevo elemento unido se guardará en el archivo “Corregir.CSV” y se creará un nuevo archivo con nombre “Unir.CSV” donde se guardará información (ver Tabla 10) de los caracter unis que serán eliminados ya que se a creado un nuevo *BBox* que contiene a los dos.

Tabla 10.

Disposición de datos en archivo Unir.CSV

Nombre de Documento PDF	**Id de fórmula**	**Coordenadas del caracter**	**Tipo de fórmula**	**Caracter**
10.1.1.1.2005_4	15	Coordenadas Xi,Xf,Yi,Yf	Emb	“p”

3.3.1.4. Sección: Fórmula con caracteres encontrados

Esta sección se la realizo con la finalidad de agilitar el proceso de corrección de los elementos ya que mediante una inspección visual se puede comparar la fórmula extraída con la fórmula dibujada, y concluir que en el caso de la Figura 44 la codificación de los caracteres en la base de datos es correcta y no es necesario realizar un barrido caracter por caracter para identificar los errores de codificación existentes.

Esta sección se habilita una vez que se haya accionado el botón "Siguiente fórmula" ya que como se ve en la Figura 44, en esta sección se despliega una imagen con todos los elementos que se han encontrado en la base de datos, estos elementos están dibujados mediante la ayuda de la librería OpenCV que permite dibujar caracteres codificados UTF-8 en coordenadas específicas, por lo tanto para que se dibujen los elementos en las posiciones correctas según la fórmula original, se adaptaron las coordenadas de los caracteres obtenidas de la base de datos para obtener una imagen final lo más similar posible a la original.

Figura 44. Sección fórmula con caracteres encontrados

Como OpenCV solo permite dibujar caracteres con codificación UTF-8, es posible que algunos caracteres especiales no se grafiquen y no se pueda realizar esta breve inspección, por lo tanto en estos casos sería necesario realizar el barrido caracter por caracter para identificar los errores de codificación que puedan existir en la fórmula.

3.3.2. Creación del ejecutable (.exe)

La finalidad de realizar este procedimiento es facilitar el uso de la herramienta creada para la corrección en sistemas operativos de Windows sin la necesidad de instalar paquetes, librerías o programas extras. Se realizó la creación de este ejecutable para el sistema operativo Windows[5] mediante el software `pyinstaller` (PyInstaller Development Team, 2005-2017) el cuál congela paquetes de Python precisamente en ejecutables multiplataforma.

El procedimiento con el cual se realizó la creación del ejecutable es el siguiente:

1. Instalar `pyinstaler` mediante comando en `cmd`. El comando que se utilizó para la instalación es:

```
pip install pyinstaller
```

[5] https://www.microsoft.com/es-es/store/b/windows

2. Una vez realizada la correcta instalación, dentro de la carpeta donde se encuentra el programa de Python, se ejecuta el comando:

```
pyinstaller programa.py
```

Después de realizar los pasos anteriores, se crearán dos carpetas: *build* y *dist*, dentro de la carpeta *dist* se encontrará varios archivos, el archivo con el nombre que para este ejemplo es "`programa`" será el ejecutable requerido.

3.4. Creación de la Base de Datos

Una vez identificados y corregidos todos los errores que pudieron haber existido mediante las descripciones mencionadas en la sección 3.3 de este documento, se procede a realizar un programa en lenguaje de Python llamado "Corregircsv.py" para corregir los archivos CSV creados anteriormente y que se encuentran descritos en la Figura 21 mediante el uso de los archivos "Corregir.csv" y "Unir.csv" creados anteriormente, paso seguido mediante un programa realizado en lenguaje Python llamado "CreateXML.py" se procederá a crear los nuevos archivos XML que tendrán la información de cada documento PDF con las correcciones realizadas.

3.4.1. Corrección archivos CSV

Para realizar las correcciones de los documentos con formato CSV obtenidos anteriormente (ver sección 3.2.), se realizó un programa escrito en lenguaje Python llamado "Corregircsv.py" el cual toma la información guardada en los archivos "Corregir.csv" y "Unir.csv" para poder analizar y procesar esta información con la finalidad de realizar los cambios pertinentes y necesarios en los archivos con formato CSV obtenidos de la base de datos.

Este programa consta con cuatro funciones principales las cuales han sido nombradas como: "`borrelem`", "`correlem`", "`corrline`" y "`nuevoelem`".

A continuación se describe la funcionalidad de cada una de las funciones mencionadas:

- Función "`borrelem`": esta función toma la información almacenada en el archivo "Unir.csv" y en primera instancia se realiza una verificación de duplicidad de elementos en caso de que existan datos duplicados, se elimina el dato más antiguo y se mantiene el más reciente, finalmente se ordena alfabéticamente todos los datos.

  Una vez realizadas las verificaciones mencionadas, mediante la información que se encuentra en el archivo "Unir.csv" (ver Tabla 10), se procede a eliminar de los archivos "Nombre_Char.csv" y "Nombre_Line.csv" ("Nombre" hace referencia a los nombres de los documentos PDF que forman parte de la base de datos) cada uno de los datos que se encuentran dentro del archivo "Unir.csv".

- Función "`correlem`": esta función utiliza la información almacenada en el archivo "Corregir.csv", de forma similar a la función "`borrelem`" se realiza una verificación de duplicidad y se ordenan los datos. Hay que tomar en cuenta que en este archivo se encuentran también datos de nuevos elementos que se van a agregar a la base de datos, por lo tanto es importante saber cuáles son nuevos elementos y cuáles son simplemente elementos que se va a corregir la codificación del carácter.

  De forma similar a la función anterior, mediante la información que se encuentra en el archivo "Corregir.csv" (ver Tabla 9), se procede a corregir la codificación de los archivos "Nombre_Char.csv" obtenidos anteriormente.

- Función "`corrline`": tiene la misma funcionalidad que la función "`correlem`" con la diferencia de que se encarga de corregir la codificación de los archivos "Nombre_Line.csv".

- Función "`nuevoelem`": esta función utiliza la información almacenada en el archivo "Corregir.csv", de igual manera que en la función "`correlem`", se realiza una verificación de duplicidad y se discrimina los datos de corrección para solo trabajar con los datos de nuevos elementos a agregar en la base de datos.

Después de realizar las verificaciones pertinentes, se procede a añadir los nuevos elementos en los archivos “Nombre_Char.csv” y “Nombre_Line.csv” obtenidos anteriormente.

3.4.2. Creación de la nueva base de datos

Debido a que se han realizado varias correcciones y se ha añadido nuevos elementos, en lugar de modificar los archivos XML pertenecientes a la base de datos de la que parte el presente trabajo (que de ahora en adelante se la llamará marmoth_data), se van a crear nuevos archivos XML con los elementos que se ha corregido y nuevos elementos añadidos.

Los nuevos archivos XML contendrán la información de las fórmulas matemáticas dentro da cada uno de las 400 paginas de documentos que se encuentran en formato PDF de la base de datos.

Por lo tanto la nueva base de datos constará de 400 páginas de documentos en formato PDF, existirá un *ground truth* de archivo XML por cada página. Estos archivos codificarán un total de 1575 fórmulas aisladas y 7907 fórmulas embebidas entre todos los documentos. La distribución de las versiones del formato PDF de los archivos originados digitalmente se puede ver en la

Tabla *11* teniendo desde la versión 1.1 hasta la versión 1.6, de igual manera en la Tabla 15 se detalla los distintos productores de PDF que se ha utilizado para generar los archivos.

Tabla 11.

Distribución de versiones de PDF

Versión de PDF	1.1	1.2	1.3	1.4	1.5	1.6	Total
Número de archivos	1	64	66	64	1	8	194

Tabla 12.

Distribución de Productores de PDF

Productores de PDF	Número de archivos
AFPL Ghostscript	15
Acrobat Distiller	66
Acrobat PDFWriter	10
ESP Ghostscript	23
GNU Ghostscript	10
MiKTeX pdfTeX	9
Dvipdfm	18
Dvips	4
PdfTeX	27
Others	12
Total	194

Esta nueva base de datos también tendrá disponibles las imágenes en formato TIF generadas a 500 ppp obtenidas de cada uno de los archivos PDF.

Cada archivo XML tendrá la estructura que se muestra en la Figura 45 con un elemento raíz llamado Page y este a su vez tendrá etiquetas anidadas que indicarán el tipo de fórmula

y los elementos que cada una contiene, estos archivos presentan un diseño de árbol, el cual como ya se mencionó anteriormente tiene jerarquía (ver Figura 14).

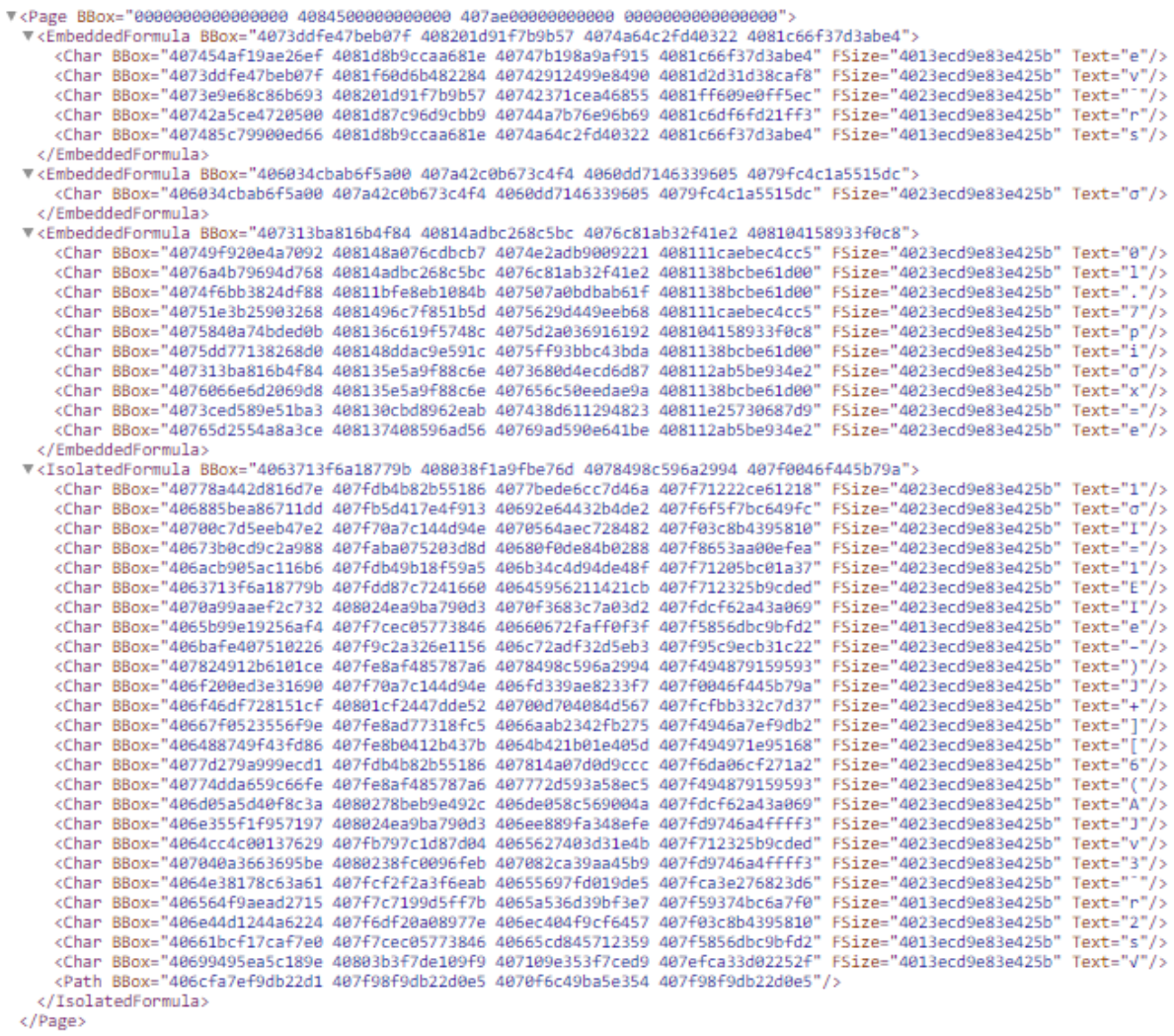


Figura 45. Estructura de los archivos XML de la nueva base de datos.

El programa encargado de realizar la nueva base de datos de archivos con formato XML llamado “CreateXML.py” está escrito en lenguaje Python el cual se lo puede encontrar en

los Anexos, y realiza el proceso que se muestra en la Figura 46, como se puede ver se toma información de cada archivo con formato CSV perteneciente a un documento y se la almacena en un archivo con formato XML el cual tendrá toda la información pertinente a las fórmulas identificadas.

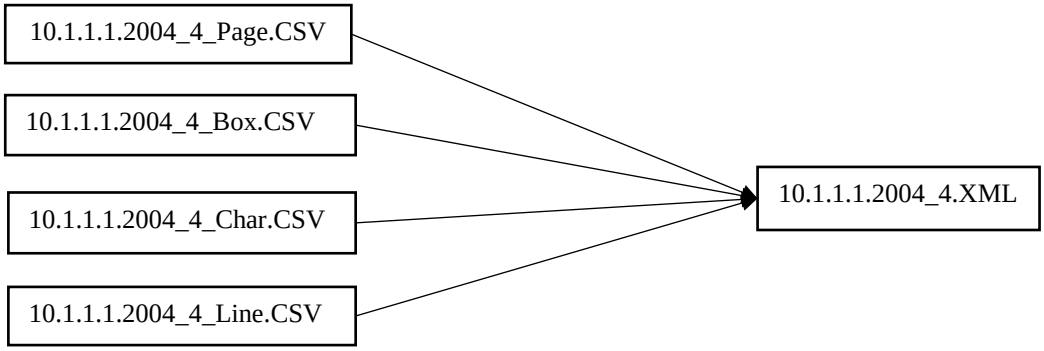


Figura 46. Diagrama de creación de archivos XML

El elemento raíz de los nuevos archivos contiene información sobre las coordenadas del tamaño total de la página, a diferencia de la base de datos con la que inicio este trabajo, en la Figura 47 se puede observar la disposición de las coordenadas, y ahora el eje Y se encuentra invertido en comparación de la base de datos marmot_data, esto se lo realizo para mayor facilidad en el manejo de las coordenadas para futuras investigaciones.

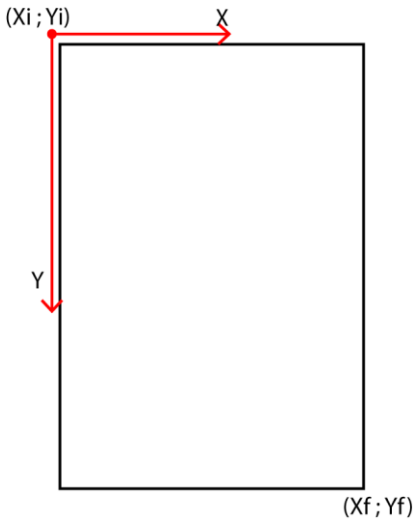


Figura 47. Descripción de coordenadas en archivos de la nueva base de datos.

Dicho elemento raíz tiene etiquetas anidadas las cuales indican el tipo de fórmula, "*EmbeddedFormula*" para fórmulas embebidas o "*IsolatedFormula*" para fórmulas aisladas, dentro de estas etiquetas existe un atributo llamado "*BBox*" que contiene las respectivas coordenadas de la fórmula.

Continuando con el modelo de jerarquía, dentro de las dos etiquetas mencionadas anteriormente, se pueden encontrar dos tipos de etiquetas anidadas llamadas "*Char*" que contiene información sobre los caracteres dentro de la fórmula o "*Path*" la cual contiene información sobre las líneas dentro de la fórmula.

Dentro de la etiqueta nombrada como "*Char*" existen tres atributos que son:

- "*BBox*": el cual contiene el atributo de las coordenadas del caracter dentro de la fórmula.
- "*Text*": contiene el atributo del tipo de caracter o símbolo en esa posición.
- "*FSize*": contiene el atributo del tamaño de fuente.

En la etiqueta "*Path*" solamente existe un atributo llamado "*BBox*" el cual contiene las coordenadas de la línea dentro de la hoja.

Cabe mencionar que las coordenadas que se han almacenado en los nuevos documentos llevan el formato hexadecimal IEEE-754 Floating-Point.

Con el procedimiento descrito se creó la nueva base de datos (que de ahora en adelante se la llamará new_marmot_data) la cual está formada por 400 archivos en formato XML los cuales contienen toda la información sobre las fórmulas extraídas de 400 documentos en formato PDF. Como ya se mencionó anteriormente esta base de datos estará disponible libremente para fines académicos e investigativos.

A continuación se detalla las diferencias existentes después de realizar las correcciones a la base de datos marmot_data.

La Tabla 13 muestra los datos del total de número de caracteres con los que se inició el presente trabajo, y el número total de caracteres que presenta la base de datos

new_marmot_data, como se puede observar existe una diferencia de 2136 elementos lo que representa un crecimiento del 1,3986 % en la base de datos new_marmot_data respecto de la base de datos marmot_data.

Tabla 13.

Comparación total de caracteres base de datos marmot_data vs base de datos new_marmot_data.

Total Caracteres (base de datos marmot_data)	Total caracteres (base de datos new_marmot_data)
152720	154856

Tabla 14.

Total de correcciones y nuevos elementos añadidos a la base de datos.

Total Correcciones	Total nuevos elementos
15643	2136

De igual manera en la Tabla 14 se detalla el número total de caracteres corregidos en la nueva base de datos, es decir se ha corregido aproximadamente el 10,2429 % del total de caracteres de la base de datos marmot_data. Entiéndase por consiguiente que la base de datos marmot_data presentaba un error del 10,2429 % en el contenido de los caracteres que se encuentran dentro de una fórmula matemática. El total de nuevos elementos que se muestra en la Tabla 14 tiene en cuenta los elementos unidos y los nuevos elementos que fueron codificados (ver Capítulo 3, sección 3.3).

Finalmente se puede establecer que la base de datos marmot_data presentaba un error total del 11,6415 %, lo cual ha sido solventado mediante: la corrección de caracteres, adición de nuevos caracteres y unión de elementos. Por lo tanto al utilizar la base de datos new_marmot_data para realizar la evaluación de algoritmos de reconocimiento de fórmulas en documentos PDF, se obtendrán resultados más reales principalmente en la evaluación de elementos que se encuentran dentro de las fórmulas.

Con las correcciones realizadas a la base de datos se tiene un *ground truth* más fiable por lo tanto permitirá que la evaluación de algoritmos de reconocimiento de fórmulas matemáticas en archivos con formato PDF sea más efectiva.

CAPITULO 4

Métodos

Una vez creada la base de datos new_marmot_data, es importante para presente trabajo validar y proveer esta información para que futuros trabajos de reconocimiento de fórmulas matemáticas en archivos PDF sean provados, por esto es importante tener una base de datos confiable con la cual realizar una evaluación del reconocimiento realizado. El siguiente capítulo describe la metodología creada con la cual se realizará la validación de la base de datos new_marmot_data.

4.1. Algoritmo de reconocimiento de fórmulas en archivos con formato PDF

Ya que es necesario obtener las coordenadas de las fórmulas y sus elementos para posteriormente poder compararlas con las coordenadas que se encuentran almacenadas en la nueva base de datos, en esta sección se detalla la manera de cómo se extrajo la información de las fórmulas matemáticas de los archivos PDF. En la Figura 48 se muestra un diagrama de bloques el cuál bosqueja el proceso que se llevará a cabo.

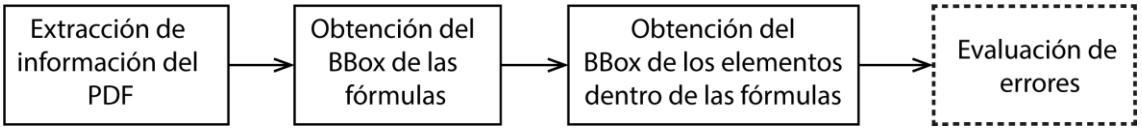


Figura 48. Diagrama de bloques del algoritmo de reconocimiento de fórmulas.

Para realizar la validación de la base de datos obtenida es necesario un algoritmo de detección de fórmulas en archivos con formato PDF, para lo cual se ha utilizado un software libre llamado *pdfminer* que es una herramienta para la extracción de información de documentos PDF. Mediante esta herramienta se puede obtener la ubicación de los caracteres en una página de un documento PDF. La información necesaria sobre la descarga e instalación de *pdfminer* se la puede encontrar en su documentación (pdfminer, 2017).

El programa *pdfminer* presenta dos herramientas que son: pdf2txt.py y dumppdf.py, para los fines necesarios en el presente trabajo se utilizó la herramienta pdf2txt.py ya que extrae información como coordenadas de posición de cada caracter, tamaño de letra del caracter, tipo de fuente de un archivo PDF, el texto representado como una cadena ASCII o Unicode.

Las opciones que presenta esta herramienta son varias y se las puede encontrar en su documentación (pdfminer, 2017), a continuación se da un ejemplo de uso de esta herramienta y se describe las opciones usadas en el presente trabajo:

Ejemplo:

```
pdf2txt.py -t xml -o Nombre.xml Nombre.pdf
```

-t tipo

Especifica el tipo de formato del archivo de salida, los formatos soportados por esta herramienta son text, html, xml, tag. Si no se especifica este campo, por defecto el

tipo de formato es text. Para el presente trabajo se utilizó el formato XML ya que la base de datos también tiene archivos XML los cuales almacenan la información de los documentos PDF.

-o nombre del documento

Especifica el nombre con el que se guardará el documento de salida. El nombre con el que se guardó cada documento de salida es el nombre perteneciente a cada documento PDF de la base de datos.

Seguido de las opciones se especifica el nombre del documento PDF del cual se desea extraer la información.

Para realizar la extracción de información de manera automática mediante el programa *pdfminer* de los 400 documentos PDF que conforman la base de datos, se realizó un programa en lenguaje Python llamado “xmlpdfminer” (ver Anexos) el cual, obteniendo el nombre de cada documento PDF ejecuta el comando necesario mostrado en el ejemplo anterior para obtener la información del texto dentro de cada documento. Por lo tanto se obtuvo 400 documentos XML que contienen la información del texto de los documentos PDF; la Figura 49 muestra el formato en el que se obtiene la información mediante el uso del programa *pdfminer*.

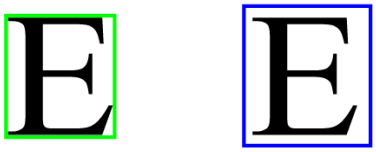
```
<?xml version="1.0" encoding="utf-8" ?>
<pages>
<page id="1" bbox="0.000,0.000,595.000,842.000" rotate="0">
<textbox id="0" bbox="73.961,748.409,291.752,773.016">
<textline bbox="73.961,759.929,291.752,773.016">
<text font="TimesNewRomanPSMT" bbox="73.961,759.929,78.941,773.016" size="13.087">k</text>
<text font="TimesNewRomanPSMT" bbox="78.760,759.929,83.183,773.016" size="13.087">e</text>
<text font="TimesNewRomanPSMT" bbox="83.201,759.929,88.181,773.016" size="13.087">y</text>
<text font="TimesNewRomanPSMT" bbox="88.001,759.929,90.491,773.016" size="13.087"> </text>
<text font="TimesNewRomanPSMT" bbox="91.721,759.929,96.701,773.016" size="13.087">p</text>
<text font="TimesNewRomanPSMT" bbox="96.642,759.929,101.622,773.016" size="13.087">o</text>
<text font="TimesNewRomanPSMT" bbox="101.562,759.929,104.878,773.016" size="13.087">r</text>
<text font="TimesNewRomanPSMT" bbox="104.802,759.929,107.571,773.016" size="13.087">t</text>
<text font="TimesNewRomanPSMT" bbox="107.455,759.929,110.224,773.016" size="13.087">i</text>
<text font="TimesNewRomanPSMT" bbox="110.108,759.929,115.088,773.016" size="13.087">o</text>
<text font="TimesNewRomanPSMT" bbox="115.148,759.929,120.128,773.016" size="13.087">n</text>
<text font="TimesNewRomanPSMT" bbox="119.948,759.929,123.822,773.016" size="13.087">s</text>
<text font="TimesNewRomanPSMT" bbox="123.666,759.929,126.156,773.016" size="13.087"> </text>
<text font="TimesNewRomanPSMT" bbox="127.386,759.929,132.366,773.016" size="13.087">o</text>
<text font="TimesNewRomanPSMT" bbox="132.426,759.929,135.742,773.016" size="13.087">f</text>
<text font="TimesNewRomanPSMT" bbox="135.545,759.929,138.035,773.016" size="13.087"> </text>
<text font="TimesNewRomanPSMT" bbox="139.265,759.929,142.582,773.016" size="13.087">r</text>
<text font="TimesNewRomanPSMT" bbox="142.505,759.929,146.928,773.016" size="13.087">e</text>
<text font="TimesNewRomanPSMT" bbox="146.812,759.929,151.234,773.016" size="13.087">a</text>
<text font="TimesNewRomanPSMT" bbox="151.119,759.929,153.888,773.016" size="13.087">l</text>
<text font="TimesNewRomanPSMT" bbox="153.772,759.929,156.262,773.016" size="13.087"> </text>
<text font="TimesNewRomanPSMT" bbox="157.492,759.929,161.914,773.016" size="13.087">a</text>
<text font="TimesNewRomanPSMT" bbox="161.799,759.929,166.779,773.016" size="13.087">p</text>
<text font="TimesNewRomanPSMT" bbox="166.719,759.929,171.699,773.016" size="13.087">p</text>
<text font="TimesNewRomanPSMT" bbox="171.639,759.929,174.408,773.016" size="13.087">l</text>
<text font="TimesNewRomanPSMT" bbox="174.401,759.929,177.170,773.016" size="13.087">i</text>
<text font="TimesNewRomanPSMT" bbox="177.055,759.929,181.477,773.016" size="13.087">c</text>
<text font="TimesNewRomanPSMT" bbox="181.361,759.929,185.784,773.016" size="13.087">a</text>
<text font="TimesNewRomanPSMT" bbox="185.668,759.929,188.437,773.016" size="13.087">t</text>
<text font="TimesNewRomanPSMT" bbox="188.321,759.929,191.090,773.016" size="13.087">i</text>
<text font="TimesNewRomanPSMT" bbox="190.975,759.929,195.955,773.016" size="13.087">o</text>
<text font="TimesNewRomanPSMT" bbox="196.014,759.929,200.994,773.016" size="13.087">n</text>
<text font="TimesNewRomanPSMT" bbox="200.814.759.929.204.689.773.016" size="13.087">s</text>
```

Figura 49. Formato archivo XML obtenido mediante el programa *pdfminer*.

Cada documento XML tiene una estructura de árbol con etiquetas anidadas, las etiquetas *text* almacenan la información de cada elemento dentro del documento como el tipo de letra en *font*, las coordenadas en *BBox*, el tamaño de letra en *size*, y el tipo de caracter o símbolo. Esta información no es solo de las fórmulas matemáticas sino de todos los elementos encontrados dentro del documento PDF.

Ya que los documentos XML obtenidos mediante la herramienta *pdfminer* contienen toda la información de cada página de los documentos PDF, es necesario realizar un proceso de discriminación de los datos que conforman una fórmula matemática de los que solo son datos de texto, para esto se realizó un programa llamado "xmlpdfminerformulas.py" (ver Anexos**¡Error! No se encuentra el origen de la referencia.**) en lenguaje Python que, tilizando la base de datos new_marmot_data, realiza el proceso de discriminación y también entrega sus datos con una estructura de árbol como se muestra en la Figura 45, a los documentos XML.

Como se puede ver en la Figura 50, debido a que el *BBox* obtenido mediante el programa *pdfminer* (Figura 50 (b), *BBox* de color azul) no es el exacto de cada caracter, como los que se tiene en la base de datos (Figura 50 (a), *BBox* de color verde), para la discriminación se realizó el siguiente proceso:



(a) (b)

Figura 50. *BBox* base de datos vs *BBox* pdfminer

(a) *BBox* en la base de datos, (b) *BBox* obtenido mediante el programa *pdfminer*.

Debido a que se desea realizar una comparación entre los dos *BBox* de la Figura 50, una forma de hacerlo es agrandado el *BBox* de la base de datos para que de esta forma las

dimensiones del *BBox* obtenido del programa *pdfminer* queden dentro del *BBox* de la base de datos new_marmot_data.

Por lo tanto a los *BBox* de cada caracter de la base de datos se agrandó sus dimensiones para posteriormente realizar un barrido dentro de los documentos XML obtenidos por el programa *pdfminer* y comparar los *BBox* para encontrar uno y solo un *BBox* pertenecientes a los documentos XML obtenidos con el programa *pdfminer* que se encuentre dentro de las dimensiones del *BBox* agrandado.

Para esto se ensancho un total de 3.3 pixels (eje x) y se alargó un total de 22 pixels (eje y) a cada *BBox*, este proceso se lo puede evidenciar en los Anexos. (línea de código: 115 a 118).

En la Figura 51, se muestra un ejemplo del proceso que se llevó a cabo para realizar la discriminación de los datos que pertenecen a una fórmula matemática frente a los que no pertenecen.

El proceso de comparación de los *BBox* realiza la siguiente lógica:

- Si el *BBox* obtenido del programa *pdfminer* se enuentra dentro de las dimensiones del *BBox* de la base de datos (que ya ha sido agrandado), entonces ese *BBox* obtenido del

programa *pdfminer* pertenece a un elemento de una fórmula matemática. Caso contrario no se los toma en cuenta para ningún análisis posterior.

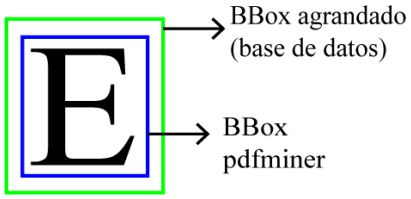


Figura 51. Proceso de discriminación

Una vez finalizada la discriminación de los elementos de cada fórmula matemática, se procede a obtener las coordenadas de la fórmula matemática que contiene a todos los elementos. En la Figura 52 se puede ver un ejemplo de este proceso, cada elemento está contenido por su respectivo *BBox*, los puntos de color rojo en los extremos superiores derechos de los *BBox* representa Xi y Yi, los puntos de color amarillo en los extremos inferior izquierdo representa Xf y Yf. Por lo que se puede observar que el Xi menor es el que pertenece al elemento con el caracter “x”, el Yi menor es el que pertenece al “3” y el Xf mayor es el que pertenece al “2”, el Yf mayor pertenece al elemento con el caracter “y”, obteniendo el *BBox* de la formula como se muestra en la Figura 53 (el *BBox* de la fórmula se encuentra con lína discontinua).

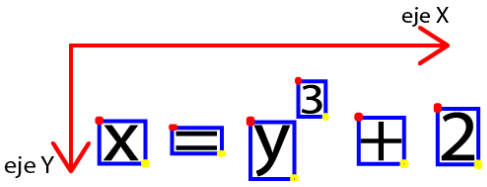


Figura 52. Ejemplo de coomparación de coordenadas de los elementos dentro de una fórmula matemática.

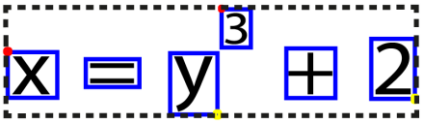


Figura 53. Ejemplo de obtencion de *BBox* de una fórmula.

Finalmente el programa agrupa la información de cada fórmula con los elementos de la misma y sus respectivas coordenadas de *BBox*, tamaño de fuente y caracter, para escribir un documento XML con una estructura jerárquica de árbol como se indicó anteriormente. La Figura 54 muestra como ha quedado alamacenada la información, con etiquetas anidadas que distinguen los tipos de fórmulas y los elementos que pertenecen a ellas.

```
▼<Page BBox="0000000000000000 4082980000000000 408a500000000000 0000000000000000">
  ▼<EmbeddedFormula BBox="407fdcd0e5604189 407adbc28f5c28f6 4080637ced916873 4079fae560418937">
     <Char BBox="4080328f5c28f5c3 407a85b645a1cac1 40804c7ae147ae14 4079fae560418937" Fsize="40215a9fbe76c8b4" Text="o"/>
     <Char BBox="40801e6666666666 407a85b645a1cac1 408032916872b021 4079fae560418937" Fsize="40215a9fbe76c8b4" Text="r"/>
     <Char BBox="4080047ced916873 407a85b645a1cac1 40801e6872b020c5 4079fae560418937" Fsize="40215a9fbe76c8b4" Text="p"/>
     <Char BBox="407fdcd0e5604189 407adbc28f5c28f6 4080048f5c28f5c3 407a07d2f1a9fbe7" Fsize="402a7e76c8b43958" Text="t"/>
     <Char BBox="40804c78d4fdf3b6 407a85b645a1cac1 4080637ced916873 4079fae560418937" Fsize="40215a9fbe76c8b4" Text="c"/>
   </EmbeddedFormula>
```

Figura 54. XML finales obtenidos del algoritmo de reconocimiento de fórmulas.

Al finalizar este proceso se obtienen 400 documentos con formato XML logrados del algoritmo de detección de fórmulas creado, que contienen la información de las fórmulas matemáticas de cada documento PDF de la base de datos, con los que posteriormente se evaluará la validez de la base de datos new_marmot_data.

Cabe mencionar que se utilizó el programa *pdfminer* y una discriminación con la cual no se puede tener una alta confiabilidad en la detección de fórmulas matemáticas ya que la finalidad de este proyecto no es desarrollar un algoritmo de detección de fórmulas matemáticas en documentos PDF, sino realizar una base de datos con la cual en futuros trabajos se pueda evaluar diferentes tipos de algoritmos de reconocimiento de fórmulas matemáticas y con esto poder tener una comparación equitativa de los diferentes tipos de algoritmos.

Una vez obtenidos los *BBox* de las fórmulas matemáticas y de sus elementos, una manera de definir si los *BBox* obtenidos en realidad pertenecen a fórmulas matemáticas, es comparar las posiciones de los *BBox* obtenidos mediante el algoritmo utilizado y los *BBox* de la base de datos new_marmot_data, si las posiciónes de los *BBox* coinciden quiere decir que efectivamente el algoritmo reconoció una fórmula matemática.

Finalmente será necesaria la evaluación del algoritmo creado mediante el uso de la base de datos new_marmot_data, a continuación se detallará métricas que servirán para evaluar

algoritmos que realizan este tipo de trabajo y de esta manera tener resultados claros para que en trabajos futuros se pueda realizar comparaciones entre diferentes tipos de algoritmos

4.2. Métrica de evaluación

Para realizar los programas de evaluación es necesario primero definir una métrica de evaluación para saber si el reconocimiento de fórmulas matemáticas tanto en posición de los *BBox* de las fórmulas y de sus elementos constitutivos son correctos, por otra parte también es importante evaluar si el caracter de cada elemento es el correcto.

A continuación se describirá los tipos de errores que se pueden presentar en la identificación de las posiciones de los *BBox,* estos tipos de errores han sido propuestos en el trabajo publicado con el nombre *Performance Evaluation of Mathematical Formula Identification* (Xiaoyan Lin L. G., 2012) donde se definen ocho tipos de situaciones que pueden presentarse.

En la Figura 55 se ilustran los diferentes tipos de errores que se pueden presentar en la identificación de fórmulas matemáticas, la región enmarcada de color verde ejemplifican los *BBox* de la base de datos y la región enmarcada de color azul ejemplifican los *BBox* de la identificación de fórmulas.

1) Correcto.- Cuando la región detectada coincide exactamente con la región de la base de datos.
2) Perdida.- Cuando no existe una región detectada y en la base de datos si existe una región.
3) Falsa.- Cuando se detecta una región y en la base de datos no existe la región detectada.
4) Parcial.- Cuando la región detectada cumple con lo siguiente: coincide parcialmente con una región de la base de datos, no está sobrepuesta sobre ningún otro tipo de región ya sea donde no hay fórmula o alguna otra fórmula ni tampoco divide otra región de una fórmula.
5) Expandida.- Cuando la región detectada cumple con lo siguiente: coincide exactamente con al menos una región de la base de datos y también cubre una región donde no hay fórmula o la región de otra fórmula.
6) Parcial y Expandida.- Cuando la región detectada cumple con lo siguiente: coincide parcialmente con una región de la base de datos y también cubre una región donde no hay fórmula o la región parcial de otra fórmula.
7) Fusionada.- Cuando la región detectada cubre más de una región dentro de la base de datos.
8) Dividida.- Cuando existe más de una región detectada para una región de la base de datos.

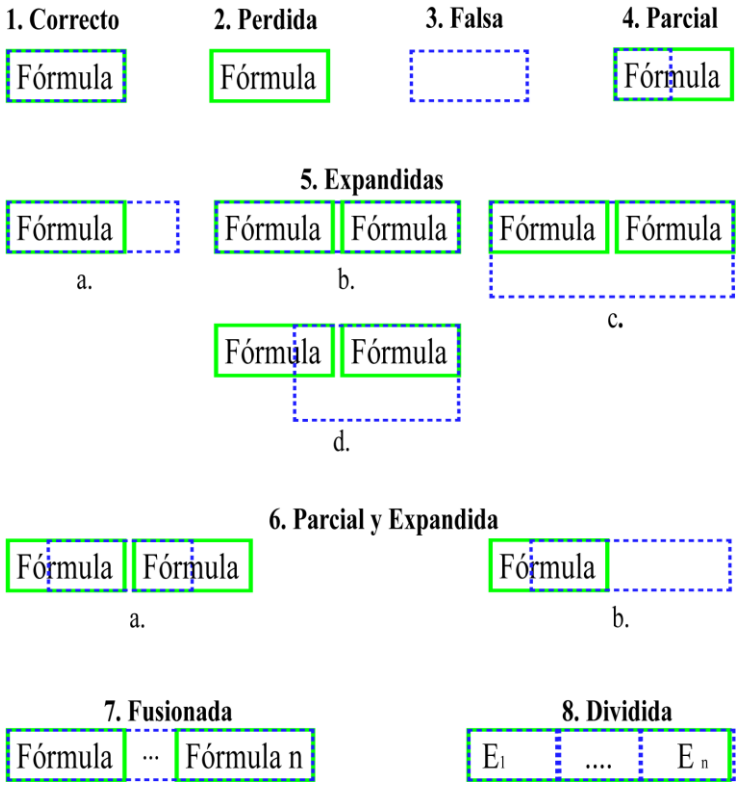


Figura 55. Tipos de errores en la identificación de *BBox* de fórmulas y *BBox* de elementos de fórmulas

Estos tipos de errores serán los que se tomará en cuenta para realizar una herramienta de evaluación final en referencia a los *BBox*.

Otro proceso que se realizará es el reconocimiento de caracteres o simbolos que posee una fórmula matemática para la evaluación de la detección de caracteres o símbolos. La Figura 56 muestra los tres tipos de situaciones que pueden presentarse, enmarcado con color verde se encuentra el carácter almacenado en la base de datos y en color azul el carácter

reconocido por el algoritmo realizado anteriormente. A continuación se describe cada situación:

1) Correcto.- Si el caracter detectado en una posición específica coincide con el de la base de datos (Figura 56 a.).
2) Incorrecto.- Si el caracter detectado en una posición específica es diferente al que se encuentra en la base de datos (Figura 56 b.).
3) Faltante.- Cuando cumple lo siguiente: si el caracter detectado no existe en la base de datos, si en la base de datos existe un caracter que no ha sido detectado (Figura 56 c.).

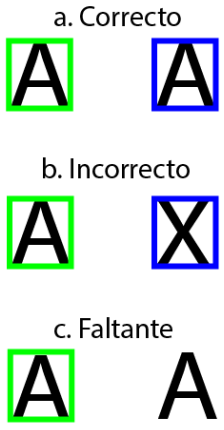


Figura 56. Tipos de situaciones en el reconocimiento de caracteres o simbolos.

4.3. Herramientas de evaluación

Para la evaluación se realizó tres programas en lenguaje de Python basados en la herramienta presentada en *Performance Evaluation of Mathematical Formula Identification* (Xiaoyan Lin L. G., 2012). Dos de los tres programas evaluaran los tipos de errores mostrados en la Figura 55 de los *BBox* tanto para las fórmulas como para los caracteres, estos programas se llaman "MathEval" (ver Anexos) y "EvalChar" (ver Anexos), mientras que el programa llamado "MathEvalCharacters" (ver Anexos) evaluará si los caracteres son los correctos. A continuación se describe cada uno de los programas mencionados.

En cada uno de estos programas se deberá especificar la dirección o *Path* de los archivos XML obtenidos del algoritmo de reconocimiento de fórmulas y la dirección del *ground thrut* que contiene los archivos XML de la base de datos.

Los archivos XML obtenidos de la detección de fórmulas que se especificarán en los programas, deben tener la estructura mostrada en la Figura 15, es decir con una estructura de árbol y especificando información sobre formulas aisladas y formulas embebidas, ya que de no ser así el programa no podrá funcionar correctamente.

4.3.1. MathEval

El programa que se describirá a continuación, evaluará los tipos de errores o situaciones que pueden presentarse en la detección de fórmulas matemáticas, este programa "MathEval" (ver Anexos) está escrito en lenguaje de Python.

Para la correcta funcionalidad de este programa es necesario especificar el *path* o dirección de la carpeta que contiene todos los archivos XML resultantes del algoritmo de reconocimiento de fórmulas y la dirección del *ground truth* donde se encuentra los archivos XML de la base de datos. Cabe recalcar que los archivos XML resultantes del algoritmo de detección de fórmulas matemáticas deben tener la estructura que se muestra en la Figura 45.

Una vez realizadas las especificaciones se procede a ejecutar el programa, esto se lo puede hacer mediante el `cmd` de Windows simplemente escribiendo: Python seguido del nombre del programa (teniendo Python como variable de entorno), ejemplo: >> Python MathEval.py

El análisis que se realiza para saber si las regiones de cada fórmula son correctas está relacionado con las coordenadas y con el área de cada región, de esta manera se podrá identificar cuál de los ocho tipos de situaciones mostradas en la Figura 55 se presenta en cada caso.

La comparación de las áreas y las coordenadas de cada región se las realiza de la siguiente manera:

- Si las áreas de los dos *BBox* son las mismas y las coordenadas son iguales entonces se concluye que la detección es correcta.

- En el caso de que las coordenadas X1;Y1 o X2;Y2 de los *BBox* (ver Figura 57) coincidan y el área de la región detectada no sea mayor al área de la región obtenida de la base de datos como se puede ver en la Figura 57, se considera una detección parcial ya que la región detectada, que se encuentra en color azul, es menor que la región de la base de datos que se encuentra de color verde. Si por otro lado cumple con la condición de que coincida una de las dos coordenadas pero el área de la región detectada es mayor al área de la región obtenida de la base de datos, como se ejemplifican los casos en la Figura 55 – 5, se considera como detección expandida.

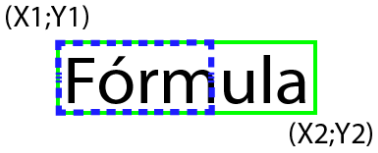


Figura 57. Detección parcial

- En caso de que las coordenadas iniciales coincidan una región y las coordenadas finales coincidan con las de otra región, y el área de la región detectada sea la suma de las áreas de las dos regiones obtenidas de la base de datos, como se ejemplifica en la Figura 55 – 7, se considera una detección fusionada.

- En el caso de que las coordenadas de X1;Y1 o X2;Y2 detectadas que forman una región rectangular se encuentren dentro de las coordenadas obtenidas de la base de

datos y que el área de la región detectada sea mayor a la del área obtenida de la base de datos, como se ejemplifica en la Figura 55 – 6, se considera una detección parcial y expandida.

- Si las coordenadas iniciales de una región detectada coinciden con las coordenadas iniciales de una región de la base de datos y también otras coordenadas iniciales de una región detectada se encuentran dentro de la misma región de la base de datos y la suma del área de las regiones detectadas es igual a el área de la región obtenida de la base de datos, tal y como se ejemplifica en la Figura 55. – 8, se considera que la detección es del tipo dividida.

- Si una de las coordenadas que se encuentra en la base de datos no coincide con ninguna coordenada detectada y ninguna de las coordenadas detectadas se encuentra dentro de las coordenadas obtenidas de la base de datos, como se ejemplifica en la Figura 55. – 2, se considera una detección faltante es decir que el programa de detección usado no encontró una región de fórmula matemática donde sí existe.

- En el caso de que las coordenadas detectadas que forman una región no coincidan o no se encuentren dentro de ninguna de las regiones formadas por las coordenadas obtenidas de la base de datos, como se ejemplifica en la Figura 55. – 3, se considera una detección falsa, es decir el programa de detección de fórmulas matemáticas obtuvo coordenadas de una región donde no existe una fórmula.

Finalmente, después de realizar el análisis descrito, se creará un archivo con formato CSV de nombres “MathEvalForm.csv” donde se especificará la cantidad de tipos de errores encontrados en la detección de fórmulas en cada uno de los documentos PDF que se encuentran en la base de datos. El formato del documento CSV que entrega el programa al finalizar la evaluación se muestra en la Tabla 15.

Tabla 15.

Disposición de datos en archivos MathEvalForm.csv y MathEvalBBoxChar.csv

Nombre del archivo PDF	N_Ecor	N_Emis	N_Efal	N_Epar	N_Eexp	N_Epae	N_Emer
10.1.1.85.6686_25.xml	0	1	2	1	8	2	0

N_Espl	N_Icor	N_Imis	N_Ifal	N_Ipar	N_Iexp	N_Ipae	N_Imer	N_Ispl	Total
0	4	0	8	2	3	0	0	0	31

Donde:

- N_Ecor: Número de fórmulas embebidas correctas.
- N_Emis: Número de fórmulas embebidas perdidas.
- N_Efal: Número de fórmulas embebidas falsas.
- N_Epar: Número de fórmulas embebidas parciales.
- N_Eexp: Número de fórmulas embebidas expandidas.
- N_Epae: Número de fórmulas embebidas parciales y expandidas.
- N_Emer: Número de fórmulas embebidas fusionadas.

- N_Espl: Número de fórmulas embebidas divididas.
- N_Icor: Número de fórmulas aisladas correctas.
- N_Imis: Número de fórmulas aisladas perdidas.
- N_Ifal: Número de fórmulas aisladas falsas.
- N_Ipar: Número de fórmulas aisladas parciales.
- N_Iexp: Número de fórmulas aisladas expandidas.
- N_Ipae: Número de fórmulas aisladas parciales y expandidas.
- N_Imer: Número de fórmulas aisladas fusionadas.
- N_Ispl: Número de fórmulas aisladas divididas.
- En la columna “Total”, se especifica el número total de fórmulas analizadas en cada uno de los documentos PDF.

4.3.2. EvalChar

El programa llamado “EvalChar” (ver Anexos) realiza funciones muy similares al programa explicado anteriormente “MathEval”, con la diferencia de que ahora se evaluará los diferentes casos de errores solo en los elementos que pertenecen a una fórmula.

De igual manera es necesario especificar el *path* o dirección de la carpeta que contiene todos los archivos XML resultantes del algoritmo de reconocimiento de fórmulas y la dirección del *ground truth* donde se encuentran los archivos XML de la base de datos. Los archivos XML resultantes del algoritmo de detección de fórmulas matemáticas deben tener

la estructura que se muestra en la Figura 45. La ejecución de esta herramienta se la realiza de la misma forma que la descrita anteriormente en MathEval.

La comparación de las áreas y coordenadas de cada región se las realiza de la misma manera que fueron explicadas anteriormente en el programa "MathEval". Una vez realizada la evaluación se obtendrá un archivo con formato CSV llamado "MathEval*BBox*Char.csv", la disposición de los datos en este archivo es igual a la disposición mostrada en la Tabla 15, con la diferencia de que N_Ecor es el número de elementos correctos dentro de las fórmulas embebidas y N_Icor es el número de elementos correctos dentro de las formulas aisladas, así para cada caso.

4.3.3. MathEvalCharacters

Esta herramienta evalúa si los caracteres detectados por el programa de detección de fórmulas matemáticas son correctos, es un programa escrito en lenguaje Python que ha sido denominado "MathEvalCharacters" (ver Anexos). Este programa no analiza regiones ni los tipos de errores descritos anteriormente, solo se enfoca en los caracteres o símbolos detectados.

De la misma forma que las anteriores herramientas descritas, es necesario especificar el *path* o dirección de la carpeta que contiene todos los archivos XML resultantes del algoritmo de reconocimiento de fórmulas y la dirección del *ground truth* donde se encuentran los

archivos XML de la base de datos. Los archivos XML resultantes del algoritmo de detección de fórmulas matemáticas deben tener la estructura que se muestra en la Figura 45. La ejecución de esta herramienta se la realiza de igual manera que la descrita anteriormente en MathEval.

Para la detección se han dispuesto 3 tipos de situaciones que se pueden presentar:

- Cuando el caracter detectado en una posición especifica es igual al caracter que se encuentra en la misma posición en la base de datos se considera que la detección es correcta. (Figura 56- a)
- Si el caracter detectado en una posición especifica es diferente al caracter que se encuentra en la misma posición obtenido de la base de datos se considera una detección errónea o incorrecta. (Figura 56- b)
- Si en la base de datos existen caracteres que no han sido detectados por la detección de fórmulas matemáticas se considera como detección faltante. (Figura 56- c)

En la Figura 56 se ejemplifican los diferentes tipos de errores en la identificación de caracteres dentro de fórmulas matemáticas, los *BBox* de color verde hacen referencia a los caracteres que se encuentran en la base de datos, los *BBox* con línea entrecortada de color azul hacen referencia a los caracteres detectados por el algoritmo.

Finalmente después de realizar la comparación, el programa crea un archivo con formato CSV llamado “MathEvalCharacters” en el cual se encuentran los datos de la evaluación realizada, se especifica la cantidad de detecciones erronas, detecciones correctas y detecciones faltantes en cada uno de los documentos PDF que se encuentran en la base de datos. En la Tabla 16, se puede ver el formato en el que se disponen de los datos de salida del programa.

Tabla 16.

Disposición de datos en archivos MathEvalCharacters.csv

Nombre de archivo PDF	Nc_Ecor	Nc_Emis	Nc_Eincor	Nc_Icor	Nc_Imis	Nc_Iincor	Total
10.1.1.6.2250_36.xml	86	5	4	198	27	5	325

Donde:

- Nc_Ecorr: Número de caracteres correctamente codificados en fórmulas embebidas.
- Nc_Emis: Número de caracteres faltantes en fórmulas embebidas.
- Nc_Eincor: Número de caracteres mal codificados en fórmulas embebidas.
- Nc_Icorr: Número de caracteres correctamente codificados en fórmulas aisladas.
- Nc_Imis: Número de caracteres faltantes en fórmulas aisladas.
- Nc_Iincor: Número de caracteres mal codificados en fórmulas aisladas.

- En la columna “Total”, se especifica el número total de fórmulas analizadas en cada uno de los documentos PDF.

CAPITULO 5

Análisis de Resultados

5.1. Resultados del uso de la nueva base de datos en la evaluación de un algoritmo de detección de fórmulas matemáticas en documentos PDF

Para realizar la evaluación de la detección de fórmulas matemáticas se utilizó como algoritmo de detección de fórmulas matemáticas en documentos PDF el método descrito en el Capítulo 4, sección 4.1.

A continuación se detalla los resultados obtenidos con las diferentes herramientas de evaluación descritas en el Capítulo 4, la evaluación se la realizó con el uso de las dos bases de datos; con la base de datos marmot_data y la base de datos new_marmot_data con la finalidad de realizar una comparación entre los resultados obtenidos y comprobar que la base de datos creada es útil para realizar análisis del funcionamiento de algoritmos relacionados con la detección y reconocimiento de fórmulas matemáticas.

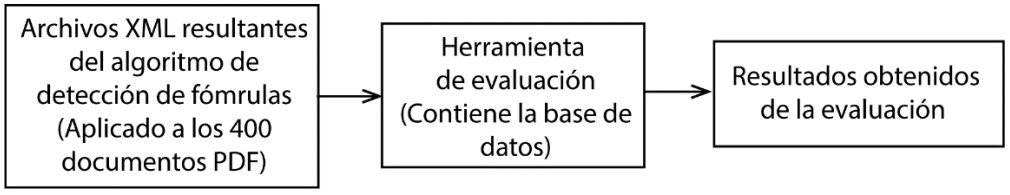


Figura 58. Diagrama del procedimiento realizado para la evaluación.

La Figura 58 muestra un diagrama del proceso realizado para la obtención de los resultados de las evaluaciones.

5.1.1. Resultados de la herramienta de evaluación MathEval

Para realizar la evaluación mediante la herramienata MathEval se ha tomado en cuenta los 400 documentos PDF que pertenecen a la base de datos. En primer lugar se aplica el algoritmo de reconocimiento de formulas descrito en el CAPITULO 4, sección 4.1 a los 400 documentos, obteniendo así los resultados que entrega dicho algoritmo. Seguido se procede a evaluar el desempeño del algoritmo mediante la herramienta MathEval utilizando la base de datos new_marmot_data, adicional, como ya se mencionó, por motivos de realizar una comparación entre los resultados, la herramienta MathEval también utilizará la base de datos marmot_data.

Una vez realizada la evaluación mediante la herramienta MathEval se han obtenido los resultados que se encuentran tabulados en la Tabla 17 y en la Tabla 18, estos resultados fueron obtenidos con el uso de la base de datos marmot_data y la base de datos new_marmot_data respectivamente.

Tabla 17.

Resultados obtenidos mediante la herramienta MathEval utilizando la base de datos marmot_data.

RESULTADOS BASE DE DATOS marmot_data			
N_Ecor	290	**N_Icor**	275
N_Emis	1851	**N_Imis**	10
N_Efal	3266	**N_Ifal**	1986
N_Epar	121	**N_Ipar**	186
N_Eexp	3960	**N_Iexp**	835
N_Epae	1619	**N_Ipae**	216
N_Emer	0	**N_Imer**	0
N_Espl	65	**N_Ispl**	106
Total		14786	

Tabla 18.

Resultados obtenidos mediante la herramienta MathEval utilizando la base de datos new_marmot_data.

RESULTADOS BASE DE DATOS new_marmot_data			
N_Ecor	290	**N_Icor**	275
N_Emis	1851	**N_Imis**	10
N_Efal	3266	**N_Ifal**	1986
N_Epar	121	**N_Ipar**	186
N_Eexp	3960	**N_Iexp**	835
N_Epae	1619	**N_Ipae**	216
N_Emer	0	**N_Imer**	0
N_Espl	65	**N_Ispl**	106
Total		14786	

- N_Ecor: Número de fórmulas embebidas correctas.

- N_Emis: Número de fórmulas embebidas perdidas.
- N_Efal: Número de fórmulas embebidas falsas.
- N_Epar: Número de fórmulas embebidas parciales.
- N_Eexp: Número de fórmulas embebidas expandidas.
- N_Epae: Número de fórmulas embebidas parciales y expandidas.
- N_Emer: Número de fórmulas embebidas fusionadas.
- N_Espl: Número de fórmulas embebidas divididas.
- N_Icor: Número de fórmulas aisladas correctas.
- N_Imis: Número de fórmulas aisladas perdidas.
- N_Ifal: Número de fórmulas aisladas falsas.
- N_Ipar: Número de fórmulas aisladas parciales.
- N_Iexp: Número de fórmulas aisladas expandidas.
- N_Ipae: Número de fórmulas aisladas parciales y expandidas.
- N_Imer: Número de fórmulas aisladas fusionadas.
- N_Ispl: Número de fórmulas aisladas divididas.

Al comparar todos los parámetros entre las dos tablas (Tabla 17 y en la Tabla 18), se observa que no existe ningún cambio en los resultados de los dos casos evaluados, es decir se obtienen los mismos resultados, esto se debe a que como se explicó en el CAPITULO 3, sección 3.3, las correcciones solo fueron realizadas en los elementos dentro de las fórmulas, mas no en los *BBox* que contienen a las fórmulas matemáticas. Por lo tanto la base de datos

se mantiene con un total de 7907 formulas embebidas y 1575 fórmulas aisladas con las que se inició el presente trabajo.

Ecuación 1. Porcentaje de reconocimientos correctos

$$\frac{N\_Ecor \ * \ 100\%}{\#Total\ de\ fórmulas\ embebidas}$$

En cuanto a los resultados obtenidos se puede verificar que utilizando la Ecuación 1 para formulas embebidas se tiene un total del 3,67% de acierto mediante el algoritmo de reconocimiento de fórmulas matemáticas usado, y para fórmulas aisladas un 17,46% de acierto.

5.1.2. Resultados de la herramienta de evaluación EvalChar

De igual manera que en la evaluación anterior, para utilizar la herramienta EvalChar en primer lugar se aplicó el algoritmo de reconocimiento de fórmulas descrito en el CAPITULO 4, sección 4.1 a los 400 documentos PDF que tiene la base de datos, una vez obtenidos los resultados del algoritmo se procede a realizar la evaluación mediante la herramienta EvalChar, esta herramienta en primera instancia utilizará la base de datos new_marmot_data, en segunda instancia utilizará la base de datos marmot_data, para posteriormente realizar una comparación con los resultados obtenidos.

Al realizar la evaluación mediante la herramienta EvalChar, se puede observar en la Tabla 19 los resultados mediante el uso de la base de datos marmot_data y Tabla 20 los resultados obtenidos mediante la base de datos new_marmot_data, por otra parte se puede realizar una comparación entre los resultados que se obtuvieron mediante la base de datos marmot_data y los resultados obtenidos al realizar la evaluación con la base de datos new_marmot_data.

Al comparar los parámetros de las dos tablas (Tabla 19 y Tabla 20), se puede observar que a pesar de que N_Ecor y N_Icor no varía al usar las dos bases de datos, pero en los diferentes casos de detecciones restantes sí existe una variación.

Tabla 19.

Resultados obtenidos mediante la herramienta "EvalChar" utilizando la base de datos marmot_data.

RESULTADOS BASE DE DATOS marmot_data			
N_Ecor	12	**N_Icor**	2
N_Emis	6041	**N_Imis**	13207
N_Efal	33777	**N_Ifal**	181186
N_Epar	0	**N_Ipar**	4
N_Eexp	15632	**N_Iexp**	44439
N_Epae	17060	**N_Ipae**	12915
N_Emer	13622	**N_Imer**	1
N_Espl	24	**N_Ispl**	7
Total		337929	

Tabla 20.

Resultados obtenidos mediante la herramienta "EvalChar" utilizando la base de datos new_marmot_data.

RESULTADOS BASE DE DATOS new_marmot_data			
N_Ecor	12	**N_Icor**	2
N_Emis	6170	**N_Imis**	13999
N_Efal	33389	**N_Ifal**	178809
N_Epar	95	**N_Ipar**	616
N_Eexp	15774	**N_Iexp**	44634
N_Epae	17238	**N_Ipae**	13126
N_Emer	13673	**N_Imer**	13
N_Espl	33	**N_Ispl**	40
Total		337623	

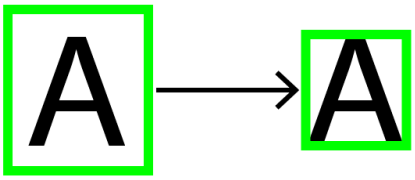
En el caso de detecciones perdidas en fórmulas embebidas (N_Emis) hay un aumento de 129 detecciones y para formulas aisladas (N_Imis) un aumento de 792 detecciones, esto indica que la base de datosnew_marmot_data presenta más caracteres que la marmot_data y que no han sido reconocidos por el algoritmo usado, esto es debido a que *pdfminer* presenta dificultad para detectar caracteres especiales que son los que en su mayoría se añadieron a la nueva base de datos.

Para el tipo de detección falsa (N_Efal y N_ifal), se tiene una disminución de 388 detecciones para fórmulas embebidas y una disminución de 2377 detecciones para fórmulas aisladas, esto indica una mejoría en la base de datos ya que al analizar estos resultados se

puede decir que la base de datos aumentó los caracteres que anteriormente fueron detectados por el algoritmo utilizado pero que no se encontraban dentro de la base de datos, esto demuestra que se ha mejorado sustancialmente la base de datos realizada en este trabajo.

Como se esperaba, los siguientes tipos de detecciones han tenido un aumento al usar la base de datos new_marmot_data, para el tipo de detección parcial (N_Epar y N_Ipar) se ha obtenido un incremento de 95 detecciones para fórmulas embebidas y de 612 detecciones para fórmulas aisladas, esto se preveía ya que en algunos casos se unieron *BBox* de caracteres que inicialmente se encontraban separados pero su correcta codificación era un solo *BBox,* a esto se debe el incremento de detecciones en este tipo de detección.

En el caso de detecciones expandidas (N_Eexp y N_Iexp) se obtuvo un aumento de 142 detecciones en fórmulas embebidas y un incremento de 195 detecciones en fórmulas aisladas, de igual manera que en el tipo de detección anterior, en algunos casos se corrigió el tamaño de los *BBox* de los caracteres, es decir si el *BBox* era demasiado grande para el caracter, se lo ha corregido con el *BBox* exacto del caracter tal y como se ejemplifica en la Figura 59.



(a) (b)

Figura 59. BBox correcto.

(a) *BBox* incorrecto (muygrande), (b) *BBox* correcto.

En cuanto al incremento de 178 detecciones en fórmulas embebidas y 211 detecciones en fórmulas aisladas para el tipo de detección parcial y expandida (N_Epae y N_Ipae) se debe a lo explicado anteriormente en el tipo de detección parcial y el tipo de detección expandida, de igual manera se han corregido los *BBox* exactos de los caracteres que era necesario hacerlo o se unieron los *BBox*.

En el caso del tipo de detección fusionada (N_Emer y N_Imer) se ha obtenido un aumento de 51 detecciones en fórmulas embebidas y un aumento de 12 detecciones en fórmulas aisladas. El tipo de detección dividida (N_Espl y N_Ispl) presenta un incremento de 9 detecciones en fórmulas embebidas y un incremento de 33 detecciones en fórmulas aisladas, de la misma manera se puede concluir que esto es debido a que se ha corregido los *BBox* de los caracteres para hacerlos lo más exactos posibles.

Como se puede ver, el incremento en los tipos de detecciones indica una mejoría en la información contenida sobre los *BBox* de los elementos que contienen las fórmulas matematicas en la base de datos new_marmot_data ya que de esta manera se puede evidenciar los cambios y correcciones realizadas para así mejorar el desempeño de las herramientas de evaluación y obtener resultados que se acerquen más a la realidad respecto a la evaluación de los elementos que se encuentran dentro de una fórmula matemática.

5.1.3. Resultados de la herramienta de evaluación MathEvalCharacters

Para utilizar la herramienta MathEvalCharacters, de igual manera que con las anteriores herramientas, en primer lugar se aplicó el algoritmo descrito en el CAPITULO 4, sección 4.1 a los 400 documentos PDF que tiene la base de datos, una vez obtenidos los resultados del algoritmo se procede a realizar la evaluación mediante la herramienta MathEvalCharacters, recordando que esta herramienta evalua si el caracter detectado es correcto respecto al de la base de datos. En primera instancia se utilizará la base de datos new_marmot_data, en segunda instancia se utilizará la base de datos marmot_data.

Al realizar la evaluación mediante la herramienta MathEvalCharacters descrita anteriormente, se puede observar en la Tabla 21 y en la Tabla 22 los resultados obtenidos, por otra parte se puede realizar una comparación entre los resultados que se obtuvieron mediante la base de datos con la que se inició este trabajo y los resultados obtenidos al realizar la evaluación con la base de datos creada en el presente trabajo.

Tabla 21.

Resultados obtenidos mediante la herramienta "EvalChar" utilizando la base de datos marmot_data.

RESULTADOS BASE DE DATOS marmot_data		%
Nc_Ecor	33661	22,041
Nc_Emis	5093	3,335
Nc_Eincor	5320	3,483
Nc_Icor	84151	55,101
Nc_Imis	10449	6,842
Nc_Iincor	14046	9,197
Total	152720	

Tabla 22.

Resultados obtenidos mediante la herramienta "EvalChar" utilizando la base de datos new_marmot_data.

RESULTADOS BASE DE DATOS new_marmot_data		%
Nc_Ecor	33611	21,705
Nc_Emis	5182	3,346
Nc_Eincor	5760	3,719
Nc_Icor	84258	54,411
Nc_Imis	10922	7,053
Nc_Iincor	15123	9,766
Total	154856	

Como se esperaba, los resultados de la evaluación de las detecciones tuvieron una variación en todos los tipos de detección que presenta la herramienta utilizada. En el tipo de

detección correcta se obtuvo una disminución de 50 detecciones para fórmulas embebidas al utilizar la base de datos new_marmot_data respecto de la base de datos marmot_data, y un incremento de 107 detecciones correctas en fórmulas aisladas, por consiguiente se tiene un 21,7% de detecciones correctas en fórmulas embebidas y un 54,41% de detecciones acertadas en fórmulas aisladas utilizando la base de datos new_marmot_data para realizar la evaluación, se puede evidenciar que el porcentaje es ligeramente menor a lo que se obtendría utilizando la base de datos marmota_data pero esto se debe a que, como se puede ver en los campos de "Total" de la Tabla 21 y la Tabla 22, se ha incrementado el número de caracteres que posee la base de datos dando paso a que el margen de error de detección de caracteres en los algoritmos de detección de fórmulas matemáticas en archivos pdf crezca.

Por consiguiente se puede verificar que el porcentaje de detecciones erróneas para cada caso, ya sea para fórmulas embebidas o para fórmulas aisladas, se ha incrementado ligeramente. Se obtuvo un incremento de 440 detecciones erróneas en fórmulas embebidas lo cual representa un aumento del 0,24% respecto del uso de la base de datos marmot_data y un incremento de 1077 detecciones erróneas en fórmulas aisladas lo que representa un 0,57% de aumento en el porcentaje respecto de utilizar la base de datos marmot_data, esto indica que al haber realizado las correcciones en la nueva base de datos que en su gran mayoría fueron de caracteres especiales, que es lo más dificultoso de detectar para los algoritmos, se mejoró el desempeño de la base de datos, obteniendo de esta manera resultados que se acercan mucho más a la realidad en el momento de evaluar un algoritmo.

De manera similar en el caso de detecciones faltantes o perdidas se puede verificar que se obtuvo un incremento de 89 detecciones en fórmulas embebidas y un aumento de 473 detecciones en fórmulas aisladas lo que representa un aumento del 0,011% y del 0,21% respectivamente utilizando la base de datos new_marmot_data frente al uso de la base de datos marmot_data, de igual manera esto demuestra que al aumentar la cantidad de caracteres dentro de la base de datos, esto también mejora el rendimiento de las herramientas de evaluación dando resultados más reales ya que en muchos casos los caracteres perdidos son caracteres especiales que anteriormente no estaban incluidos.

Por otra parte al realizar una comparación de los resultados obtenidos con la base de datos new_marmot_data frente a la base de datos marmot_data, se puede concluir que los resultados obtenidos mediante las herramientas de evaluación han mejorado en cuanto se refiere a elementos dentro de las distintas fórmulas matemáticas.

Por los resultados obtenidos podemos evidenciar que la base de datos propuesta ofrece información necesaria para evaluar algoritmos relacionados a la detección de fórmulas matemáticas en archivos PDF; además se puede verificar la validez de las herramientas de evaluación propuestas para el uso de la base de datos, obteniendo resultados con porcentajes esperados según el tipo de detección que se ha propuesto.

CAPITULO 6

Conclusiones y Recomendaciones

6.1. Conclusiones

- Una vez realizada la investigación del estado del arte, se ha concluido que en la actualidad no se puede encontrar con facilidad una base de datos que contenga una cantidad de datos representativos y válidos para realizar una evaluación fiable de algoritmos de reconocimiento de fórmulas en documentos PDF debido a que no existen o son bases de datos propietarias, lo cual dificulta su fácil uso en herramientas de evaluación que puedan comparar la validez o fiabilidad de dichos algoritmos.

- En el presente trabajo de investigación, a pesar de que se cimentó en la base de datos presentada en el trabajo *Performance Evaluation of Mathematical Formula Identification*, se ha realizado una gran cantidad de correcciones y cambios sustanciales de gran valor, principalmente en los elementos que contienen una fórmula matemática, mediante el uso de las herramientas de evaluación presentadas en el CAPITULO 4 se demuestra que esta base de

datos puede ser utilizada en la evaluación de futuros trabajos de detección de fórmulas matemáticas en archivos PDF.

- En esta investigación se ha creado una base de datos que contiene 400 hojas de documentos PDF dentro de las cuales se encuentran 9482 fórmulas matemáticas, 7907 embebidas y 1575 aisladas, por lo tanto después del análisis realizado se puede establecer que los datos que contiene la nueva base de datos presentada (new_marmot_data) es un conjunto de datos representativo de los documentos en el mundo real.

- Mediante la utilización de técnicas de procesamiento digital de imágenes PDI, principalmente mediante la técnica de componentes conexos y con la ayuda de librerías especializadas en PDI para lenguaje Python como OpenCV, se ha podido reconocer, codificar y añadir una gran cantidad de elementos a la base de datos para de esta manera ampliar la información y mejorar el rendimiento de la base de datos presentada.

- Se ha presentado tres herramientas de evaluación de algoritmos de reconocimiento de fórmulas matemáticas en archivos PDF, el primero evalúa la detección de fórmulas dentro de un documento PDF, el segundo evalúa la detección de elementos dentro de una fórmula matemática, la tercera evalúa los tipos de caracteres reconocidos dentro de la fórmula matemática, cada una

entrega un archivo detallando los tipos de detecciones encontrados, esto con la finalidad de poder realizar una rápida comparación entre algoritmos.

- Para la evaluación se ha mantenido la métrica propuesta en el trabajo en el que se basa esta investigación (*Performance Evaluation of Mathematical Formula Identification*) ya que después de un análisis detenido se concluyó que esta métrica abarca todas las posibles detecciones de un algoritmo de detección de fórmulas dentro de un archivo PDF. Por otra parte, se adaptó esta metrica para la evaluación de los caracteres reconocidos dentro de una fórmula matemática y se propuso tres posibles detecciones que son: correcta, incorrecta y faltante o perdida.

- Se ha probado la validez de la base de datos propuesta mediante las tres herramientas de evaluación presentadas en este trabajo, obteniendo resultados esperados para cada caso de detección confirmando así la validez de la base de datos.

- La base de datos y las tres herramientas de evaluación propuestas en este trabajo serán publicadas de forma gratuita para fines investigativos y educativos.

6.2. Recomendaciones y trabajos futuros

- Para utilizar las herramientas de evaluación, se recomienda tomar en cuenta el formato de árbol y las etiquetas específicas del archivo XML que ingresará a ser evaluado tal como han sido descritas en este documento, ya que de no ser así, la evaluación no se realizará de forma correcta.

- En caso de utilizar la herramienta con la que se llevó a cabo la corrección y adición de nuevos elementos a la base de datos, es recomendable entender las indicaciones para tener en cuenta la utilidad de cada sección de esta herramienta, caso contrario se puede realizar correcciones de manera incorrecta lo que podría generar dificultades.

- Como trabajo futuro se recomienda ampliar la cantidad de hojas de archivos PDF que contiene la base de datos, tomando en cuenta las nuevas versiones de PDF, para de esta manera mejorar su desempeño.

- Mediante el uso de la base de datos propuesta, a futuro se puede mejorar o crear un sistema de reconocimiento de fórmulas matemáticas en archivos PDF y ser evaluado con las herramientas propuestas.

- A futuro se puede crear aplicaciones para personas no videntes que requieran del acceso a documentos con fórmulas matemáticas y de esta manera aportar a la inclusión de personas con discapacidad.

BIBLIOGRAFÍA

Adobe Systems Incorporated. (2006). *PDF Reference.*

Asamblea Constituyente. (Octubre de 2008). Constitución de la República del Ecuador. *Registro Oficial No.449*.

Asamblea Nacional del Ecuador. (25 de Septiembre de 2012). Ley Orgánica de Discapacidades.

Garain, U. (2009). Identification of Mathematical Expressions in Document Images. *International Conference on Document Analysis and Recognition*.

Jianming Jin, X. H. (2003). Mathematical Formulas Extraction . International Conference on Document Analysis and Recognition.

Josef B. Baker, A. P. (2009). A Linear Grammar Approach to Mathematical Formula Recognition from PDF. *School of Computer Science, University of Birmingham* .

K. Ashida, M. O. (2006). Performance evaluation of a math formula recognition system with a large scale of printed formula images. *Document Image Analysis for Libraries*.

Luis Miralles Pechuán, D. R. (2015). Reconocimiento de dígitos escritos a mano mediante métodos de tratamiento de imagen y modelos de clasificación.

M. Suzuki, S. U. (2005). A ground-truthed mathematical character and symbol image database. *Proc. Int. Conf. Document Analysis and Recognition.*

Marisa R. De Giusti, M. M. (2005). MANUSCRIPT DOCUMENT DIGITALIZATION AND RECOGNITION: A FIRST APPROACH. *JCS&T Vol. 5 No. 3*.

Marisa R. De Giusti, M. M. (s.f.). Uso de componentes conexas para restauracion automatica de documentos digitalizados. *Comisión de Investigaciones Científicas de la Provincia de Buenos Aires*.

Ministerio de Salud Pública. (s.f.). *MSP*. Recuperado el 7 de Septiembre de 2017, de http://www.salud.gob.ec/calificacion-o-recalificacion-de-personas-con-discapacidad-2/

OMS. (12 de Julio de 2017). *Organización Mundial de la Salud*. Obtenido de http://www.who.int/topics/disabilities/es/

OpenCV. (22 de 12 de 2017). *Open Source Computer Vision*. Obtenido de https://docs.opencv.org/3.4.0/dd/d49/tutorial_py_contour_features.html

pdfminer. (18 de 11 de 2017). *pdfminer Release 0.0.1*. Obtenido de https://media.readthedocs.org/pdf/pdfminer-docs/latest/pdfminer-docs.pdf

PyInstaller Development Team. (2005-2017). *PyInstaller*. Obtenido de http://www.pyinstaller.org/

Python. (2017). *Python*. Obtenido de https://docs.python.org/2/library/struct.html

Torres, A. D. (2015). *Red de Revistas Científicas de América Latina y el Caribe, España y Portugal*. Obtenido de UAEM redalyc.org: http://www.redalyc.org/html/132/13207206/

U. Garain, a. B. (2005). A corpus for OCR research on mathematical expressions vol.7. *Document Analysis and Recognition*.

Utpal Garain, B. B. (2004). Identification of Embedded Mathematical Expressions in Scanned Documents. *International Conference on Pattern Recognition (ICPR'04)*.

UW-III, W. U. (1996). English/Technical Document Image Database.

Xiaoyan Lin, L. G. (2011). Mathematical Formula Identification in PDF Documents . *International Conference on Document Analysis and Recognition*.

Xiaoyan Lin, L. G. (2012). Performance Evaluation of Mathematical Formula Identification . *10th IAPR International Workshop on Document Analysis Systems*.

Xiaoyan Lin, L. G. (2013). Mathematical formula identification and performance evaluation in PDF documents. *Springer-Verlag Berlin Heidelberg*.