Reconocimiento de entidades relacionadas con el proceso de síntesis de kesteritas mediante herramientas de procesamiento de lenguaje natural

Natalia E. Jiménez Ordóñez, Sergio A. Perdomo Camargo
Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones
Universidad Industrial de Santander
Bucaramanga, Colombia
natalia.jimenez@correo.uis.edu.co
sergio.perdomo@correo.uis.edu.co

***Resumen*—En este artículo se describe el desarrollo de un sistema automático de reconocimiento de entidades relacionadas con la elaboración de películas delgadas basadas en kesterita, mediante diferentes métodos de síntesis para su empleo en dispositivos fotovoltaicos. Este sistema se desarrolló usando *Natural Language Processing Tool Kit (NTLK)* y Regresión Logística (RL) como técnica de aprendizaje automático, sobre una base de datos de artículos científicos encontrados en SCIENCEDIRECT. Con el objetivo de comparar los resultados, se ha aplicado un algoritmo de aprendizaje más: el algoritmo SVM(*Support Vector Machine*), con el propósito de comprobar la eficiencia y efectividad del algoritmo RL en la tarea concreta de clasificación de entidades. Lo anterior tiene como finalidad mostrar un sistema de reconocimiento de entidades que realiza la extracción, clasificación y representación de seis parámetros definidos.**

***Index Terms*—Clasificación automática de texto, Kesterita, *Natural Language Processing ToolKit de python (NLTK)*, Aprendizaje Automático, Aprendizaje Supervisado, Regresión Logística (RL), SVM (*Support Vector Machine*)**

I. INTRODUCCIÓN

El acervo de conocimiento científico producto de actividades de investigación es enorme y continúa creciendo cada día. A pesar de su importancia, esta información no puede utilizarse fácilmente debido a que se encuentra dispersa en un gran volumen de datos. Por esta razón, el tratar de procesarla manualmente resultaría en un proceso lento, costoso e ineficiente en términos de cantidad de recursos humanos. Por tal razón, se necesitan sistemas automáticos que permitan procesar, analizar y extraer información relevante de manera eficiente [1]. Muchas son las áreas de la investigación científica que se verían beneficiadas de la implementación de este tipo de sistemas, tal es el caso de la energía solar fotovoltaica, y más específicamente la síntesis de materiales en celdas solares para la generación de electricidad. Debido a lo novedoso de esta técnica, numerosos estudios de investigación se realizan actualmente para determinar las características ópticas de los materiales a usar, y de esta manera obtener una eficiencia de celda tan alta como sea posible [2]. Es allí donde se hace evidente la necesidad de una herramienta que clasifique la información recopilada.

Actualmente, la fabricación de módulos solares está basada en dos tecnologías: la primera es la denominada tecnología de Silicio mono– y policristalino o de primera generación, y la segunda (mencionada en el párrafo anterior) es la denominada tecnología de películas delgadas o de segunda generación, la cual permite fabricar módulos fotovoltaicos a costos significativamente más bajos que los de silicio [3]. Distintos materiales han sido utilizados para la elaboración de estas películas, y su aplicación depende de variables como: *compound*, *efficiency*, *band gap*, *synthesis method*, *substrate temperature*, *annealing*, *coefficient absorption*; lo cual ha centrado la atención principalmente en un material denominado kesterita. Las kesteritas son materiales semiconductores constituidos por elementos menos tóxicos y más abundantes en la naturaleza que otros similares utilizados en celdas de película delgada [4]. Cuentan con propiedades adecuadas para utilizarse como material absorbente en este tipo de celdas, con una eficiencia que puede incrementar hasta un 32 % según lo previsto teóricamente por el límite de *Shockley-Queisser(SQ)* [5] y experimentalmente a 12.6 % [6].
La tendencia en este tipo de materiales es a desarrollarse de manera empírica y atendiendo criterios subjetivos, lo que se refleja en la generación de costos adicionales durante su implementación. Sin embargo, consultando la información suministrada en base a experimentos realizados alrededor del mundo, se puede tomar ventaja de los más de tres mil artículos reportados en diferentes bases de datos de literatura internacional científica (SCIENCEDIRECT, SCOPUS Y IEEE). Todo lo anterior con el fin de mejorar su desempeño y lograr una mejor aplicación con un menor costo.

El presente trabajo busca extraer, clasificar y representar de manera automática datos de interés relacionados con el proceso de síntesis de celdas solares basadas en kesteritas. Con esta herramienta el usuario podrá analizar las variadas configuraciones de síntesis con sus respectivos resultados, y así mejorar el desempeño de futuras celdas solares.

Para lograr el objetivo planteado se diseñó un sistema que consiste en cuatro procesos: segmentación, verificación, clasificación y representación. En la segmentación y verificación se realiza el etiquetado de las entidades en seis categorías: *efficiency*, *band gap*, *synthesis method*, *substrate temperature*, *annealing*, *coefficient absorption*. Para ambos procesos se empleó *Natural Language Processing Tool Kit (NLTK)*. La clasificación realiza la elección del contenido bueno o malo de la segmentación. Para este proceso se realizaron dos algoritmos de aprendizaje automático, y se compararon por medio de una validación y finalmente se elige el de mayor rendimiento. La representación se desarrolló exportando los resultados obtenidos a una hoja de calculo de Excel, la cual permitió una mejor visualización para el usuario.

Este artículo está organizado de la siguiente manera: La sección I corresponde a la introducción, la sección II describe los datos utilizados, en la sección III se explicará la descripción del sistema, posteriormente en la sección IV se mostrará el método que se utilizó para el diseño del sistema de reconocimiento de entidades, seguidamente en la sección V se mostrará el resultado final del sistema; finalmente en la sección VI y VII se presentan las conclusiones del trabajo y se presentan recomendaciones para trabajos futuros.

II. DESCRIPCIÓN DE LA INFORMACIÓN

La materia prima del sistema corresponde a artículos recientes de diseño de celdas solares basadas en kesteritas, tomados de la base de datos *sciencedirect*. Adicionalmente, se aclara que la presente herramienta busca determinar algunos parámetros de diseño y eficiencia a partir de esta materia prima, pero de manera automática, mediante el uso de técnicas y herramientas del estado del arte. En particular, se busca determinar automáticamente parámetros tales como: *efficiency*, *band gap*, *synthesis method*, *coefficient absorption*,*substrate temperature* y *annealing*. Estas seis variables representan características fundamentales presentes en buena parte de los artículos científicos relacionados con la materia. Si bien éstas no son todas la variables implicadas en el proceso de diseño de celdas, realizar una aplicación para todas las variables necesarias es algo que está fuera del alcance del presente trabajo.

En total, se realizó la lectura de 200 artículos científicos relacionados con el diseño de celdas solares basados en kesteritas. Con éstos, se creó el conjunto de datos, guardados en formato **.txt**, y que acompaña el presente trabajo a modo de material suplementario.

III. DESCRIPCIÓN DEL SISTEMA

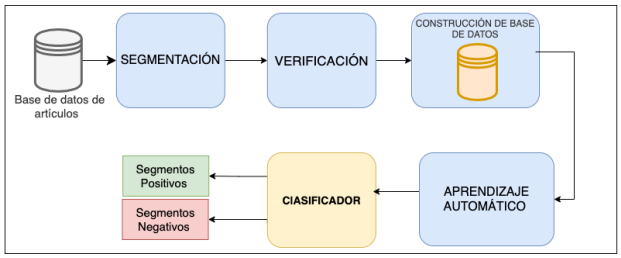

Figura 1: Diagrama de bloques del proceso de entrenamiento del sistema

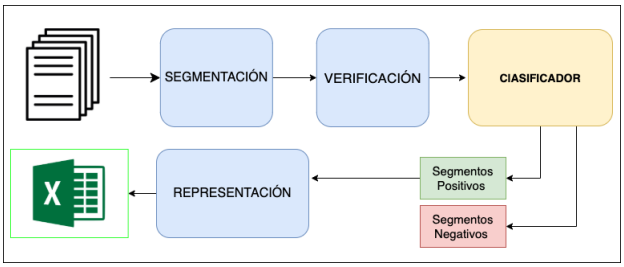

Figura 2: Diagrama de bloques del sistema funcional automático de reconocimiento de entidades

Tabla I: Parámetros

Parámetros a extraer	
Variable de Interés	**Término clave**
DOI	doi-ddi
Titulo de Paper	nombre del archivo.txt
Efficiency	compuesto + número + %
Band Gap(Eg)	compuesto + número + eV
Synthesis Method	corpus de métodos seleccionados
Sustrate Temperature	número + °C
Annealing	Annealed + °C + seg/min/días/años
Coefficient Absorption	número + Cm-1

[a] Para identificar de manera correcta cada una de las variables, es necesario establecer los parámetros que componen esta variable, por lo que el contenido de la tabla I muestra de manera resumida estos parámetros.

El sistema desarrollado se implementó usando Python. Una limitante de este lenguaje de programación es que solo trabaja con archivos en formato **.txt**(texto plano). Por esta razón, se tuvo que realizar la conversión de los archivos en formato **.pdf** (obtenidos directamente de la base de datos *sciencedirect*) a formato **.txt**. Las herramientas diseñadas para este proceso de conversión deben tener en cuenta la codificación con la que fue elaborado el documento y no siempre se obtienen los resultados esperados. Esto debido a la ocurrencia de caracteres indeseados procedentes del proceso de transformación del formato. Con el propósito de reducir lo máximo posible la ocurrencia de estos caracteres, se utilizó la herramienta en linea ”pdf2go”que cumplirá el propósito de transformación de formato.

Los procesos utilizados para el desarrollo del sistema son:

Segmentación: El proceso de segmentación separa del texto cada párrafo en donde se encuentren dichos términos claves.

Verificación: El proceso de verificación realiza un filtro, el cual comprueba si cada uno de los párrafos correspondientes a las variables extraídas del texto cumple con las características de cada variable de interés, es decir, que la información extraída cumpla con el término clave de cada una de éstas, tal como muestra la Tabla I.

Tanto la segmentación como la verificación se realizan mediante algoritmos de procesamiento natural de lenguaje (NLTK).

Clasificación: El proceso de clasificación es el encargado de determinar si los párrafos previamente extraídos de cada variable corresponden a resultados experimentales procedentes de un trabajo realizado en dicho artículo o son párrafos que referencian el trabajo de otros autores. A fin de desarrollar este proceso se aplican técnicas de aprendizaje automático.

Representación: Este proceso corresponde a la manera en que el usuario podrá visualizar los resultados, la manera más eficiente para ello es en formato Excel en donde se podrá visualizar cada uno de los resultados experimentales obtenidos en el desarrollo de cada uno de los artículos científicos que alimentan al programa, además de su fecha de publicación.

IV. MÉTODO

IV-A. Segmentación

El proceso de segmentación es el encargado de etiquetar y dividir el texto en pequeñas listas (vectores de datos), las cuales son definidas como variables de interés y se determinan por nombre y unidad (término clave). Este proceso se realiza por medio de un algoritmo de procesamiento natural de lenguaje (NLTK). Esta herramienta es una de las bibliotecas de procesamiento de lenguaje natural más conocidas y utilizadas en el ecosistema de python. NLTK es útil para varios tipos de tareas, desde tokenización hasta derivación y etiquetado, entre otras.

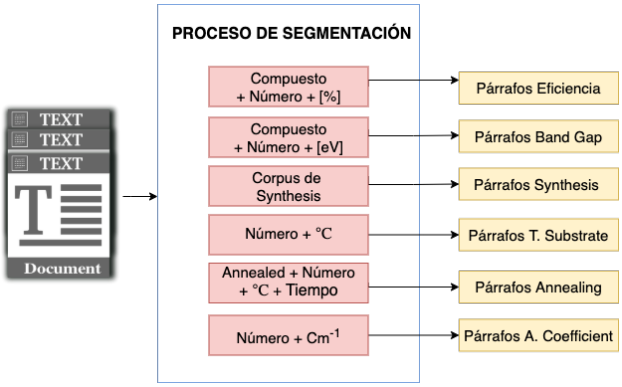

Figura 3: Diagrama de bloques del proceso de segmentación

El algoritmo ejecutó las siguientes tareas para cada una de las seis variables de interés, con el fin de realizar la segmentación del texto:

- **Tokenización:** Este procedimiento divide el texto jerárquicamente en párrafos, oraciones y palabras, utilizando caracteres delimitadores propios del lenguaje, como lo son los signos de puntuación. Este procedimiento es utilizado para la construcción de la clasificación de texto.
- **Análisis léxico:** Una vez se obtuvo la Tokenización del texto en palabras, se procedió a realizar el análisis de cada palabra con respecto al término clave de cada variable de interés. Para esto, se utilizó el reconocimiento de entidades *NER* de la librería NLTK de python, la cual tiene la función de agrupar determinadas palabras con el propósito de encontrar una relación semántica entre ellas.
- **Segmentación:** Una vez realizado el reconocimiento de entidades y como resultado del análisis léxico, se organizó cada párrafo que contiene las palabras clave de interés para cada variable en una lista.

El resultado de la ejecución de estos tres procesos es la creación de seis listas de variables de interés (una para cada variable), cada una con un contenido de información relevante a los términos claves vistos en la Tabla I.

IV-B. Proceso de Verificación

El proceso de verificación visto en el diagrama de bloques de la Fig. 3 se desarrolla por medio de un filtro, el cual elimina los párrafos en los cuales no se nombra el término clave ni la variable de interés. De esta manera se permite que sólo pasen los párrafos que contengan por completo la variable de interés más el término clave; comprobando así que cada variable contenga la información del criterio establecido en la segmentación.

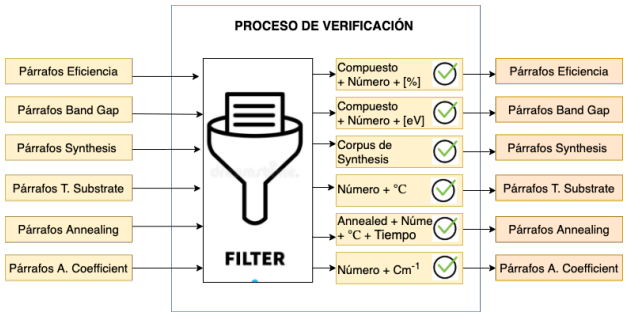

Figura 4: Diagrama de bloques del proceso de verificación

IV-C. Proceso de Clasificación

Con la finalidad de clasificar de manera positiva los párrafos provenientes de las etapas iniciales de nuestro proceso, se determinaron las siguientes condiciones: 1. El párrafo debe referirse a información que fue desarrollada por los autores en el escrito y no ser una referencia a un trabajo realizado por otros autores. 2. El párrafo debe contener

resultados experimentales referentes a las variables de interés al igual que las unidades que esta variable represente, como muestra la Tabla I.

Con este propósito se desarrolló el proceso de clasificación, en donde por medio de una selección manual de párrafos, determinada por las dos condiciones mencionadas anteriormente, se construyó e implementó una nueva base de datos, la cual contendrá los párrafos positivos y negativos de 200 artículos. Y a partir de éstos, se entrenaron dos algoritmos de aprendizaje automático con los métodos de Regresión Logística y Máquinas de vectores de soporte (SVM), los cuales se evaluaron y compararon para elegir el más adecuado a implementar en el sistema de reconocimiento de entidades.

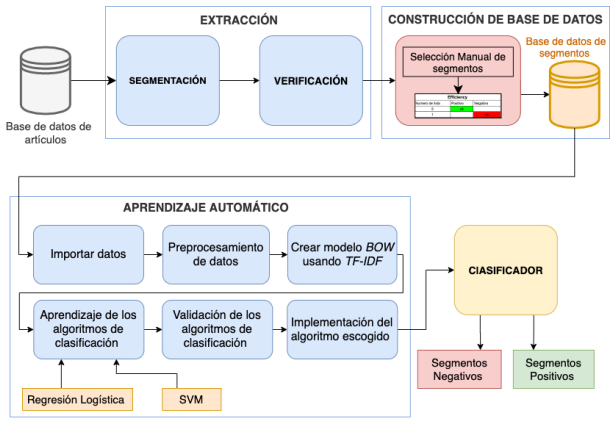

Figura 5: Diagrama de bloques del proceso de clasificación

Para realizar el proceso de clasificación visto en el diagrama de bloques de la Fig 5, se desarrollaron tres sub-procesos: Extracción, Construcción de base de datos y Aprendizaje Automático. Estos se explican a continuación.

1. **Extracción:** Se realizó la segmentación y limpieza para los 200 artículos.
2. **Selección de párrafos:** Una vez obtenida la extracción de cada uno de los 200 artículos, los párrafos se clasificaron manualmente en positivos y negativos.
3. **Aprendizaje Automático (*Machine Learning*):**

3.1 Importación de datos:

Se utilizó la función *load files* de la librería *sklearn datasets* [7] para importar el conjunto de datos al programa. Esta función divide los datos automáticamente en conjuntos de datos y categorías (clases); es decir, carga los datos de los párrafos positivos y negativos en una variable (X) y los etiqueta en categorías en otra variable (y) como 0 y 1, en donde el 0 hace referencia a los positivos y el 1 a los negativos.

3.2 Pre-procesamiento de datos:

Teniendo los datos importados se realizó el procesamiento del texto, el cual consistió en la eliminación de cuatro tipos de caracteres: caracteres especiales, espacios no deseados, caracteres individuales sin relevancia para el presente caso de estudio, y residuos del proceso de lematización (reducción de una palabra a su raíz). Ejemplos del tercer y cuarto caso serían, respectivamente: artículos y preposiciones (el, la, a, ante, etc); y la reducción, por ejemplo, de "perros" a "perro". Esta se realiza para evitar crear dobles palabras que tienen el mismo significado.

3.2 Creación del modelo utilizando bolsa de palabras y TF-IDF:

Los archivos de texto son series de palabras (ordenadas) comprensibles para los humanos, pero no para las computadoras. A diferencia de los humanos, éstas no pueden entender el texto sin formato, es decir, para ejecutar algoritmos de aprendizaje automático, es necesario convertir los archivos de texto en vectores de características numéricas. Para tal fin, se utilizó el modelo de bolsa de palabras y TF-IDF.

El modelo de bolsa de palabras *BOW*: consiste en segmentar cada archivo de texto en palabras. Una vez dividido, se realiza un conteo del número de veces que aparece cada palabra en cada uno de los archivos de texto (frecuencia) y finalmente se asigna a cada palabra una identificación de un número entero, formando un vector de palabras únicas en el diccionario que harán referencia a una característica (característica descriptiva). Este enfoque asigna una puntación a una palabra en función de su frecuencia en un texto en particular, pero no tiene en cuenta el hecho de que la palabra también podría tener una alta frecuencia de aparición en otros textos (documentos). Por esta razón, la creación del modelo va acompañada del valor TF-IDF, cuyo objetivo es reducir el impacto de las palabras que aparecen más frecuentemente en un corpus (base de datos de artículos) dado y que, por lo tanto, son evidentemente menos informativos que las palabras que se repiten en una pequeña fracción del corpus de entrenamiento [8].

TF-IDF: Es una estadística numérica que refleja la importancia de una palabra para un documento en una colección o corpus. Su cálculo se efectúa una vez el texto del documento ha sido convertido de texto a características numéricas. Todas las palabras tienen la misma importancia y no se conserva la información semántica [9].

Tabla II: Factor TF [9]

FACTOR TF	
Denominación	Frecuencia de aparición del término
Descripción	Es la frecuencia de aparición de un término a lo largo de un documento. Dicho de otra forma, el número de veces que este se repite en el documento, lo que permite determinar su capacidad de representación.
Finalidad	Representativa
Casos	- Frecuencia de aparición TF baja: Representatividad elevada. - Frecuencia de aparición TF media. - Frecuencia de aparición TF alta: Representatividad muy baja.

$$TF = \frac{Frecuencia\ de\ la\ palabra\ en\ el\ texto}{No.\ total\ de\ palabras\ en\ el\ texto} \quad (1)$$

Tabla III: Factor IDF [9]

FACTOR IDF	
Denominación	Frecuencia Inversa del Documento para un término
Descripción	Es el coeficiente que determina la capacidad discriminatoria del término de un documento con respecto a la colección. Es decir, distinguir la homogeneidad o heterogeneidad del documento a través de sus términos.
Finalidad	Discriminatoria
Casos	-Poder discriminatorio bajo: El término es genérico y aparece en la mayoría de los docs. -Poder discriminatorio medio. -Poder discriminatorio alto: El término es especializado y aparece en pocos docs.

$$IDF = log(\frac{No.\ de\ textos}{No.\ de\ textos\ que\ contienen\ la\ palabra}) \quad (2)$$

Valor TF-IDF: La relevancia de un término o palabra en un documento es el producto de su frecuencia de aparición en dicho documento (TF) y su frecuencia inversa de documento (IDF) tal como se muestra en la Fig. 6.

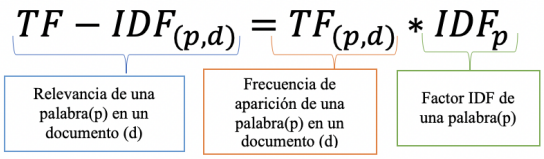

Figura 6: Valor TF-IDF para un término en un documento.

Finalmente se convierten los valores del modelo de bolsa de palabras (*BOW*) a valores TF-IDF, lo cual significa que el valor TF-IDF se calcula para cada palabra en cada documento. Los valores obtenidos son denotativos de la importancia del término en cada documento, es decir, en cuanto mayor sea el valor (puntaje) TF-IDF, más extraña será la palabra y viceversa. Este valor o puntaje se aplicará para la posterior clasificación por medio de los algoritmos supervisados de aprendizaje automático.

3.4 Aprendizaje de los algoritmos de clasificación:

Una vez obtenidos los valores TF-IDF se procede a realizar el entrenamiento de los algoritmos de clasificación. Para el presente caso se utilizó la regresión logística (RL) y máquinas de vectores de soporte (SVM) como métodos de clasificación.
A continuación se explicará brevemente cada uno de ellos, acompañado de su desarrollo.

3.4.1 Métodos de clasificación

-**Regresión logística (RL):** El análisis de regresión logística es una técnica estadística multivariable, la cual permite estudiar la relación entre una o más variables independientes y una variable dependiente de tipo dicotómica, que representa la ocurrencia o no de un suceso, por ejemplo: muerte o vida, sano o enfermo, fumador o no fumador, madre adolescente o madre no adolescente, hipertenso o no hipertenso, etc [10].

Este modelo tiene la misma estrategia que el Análisis de regresión lineal múltiple, el cual se diferencia esencialmente del análisis de regresión logística en que la variable dependiente es métrica; en la práctica el uso de ambas técnicas tiene mucha semejanza, aunque sus enfoques matemáticos son diferentes. La variable dependiente o respuesta no es continua sino discreta (generalmente toma valores 1 y 0) [10].

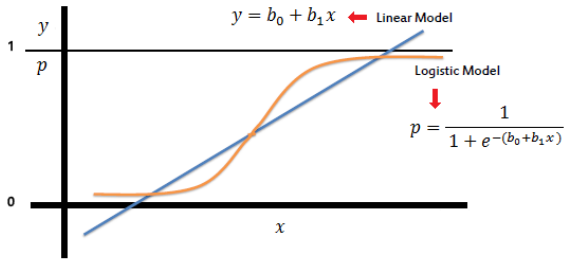

Figura 7: Modelo de regresión logística y lineal [11]

-**Máquinas de vectores de soporte (SVM):**

El algoritmo SVM (Support Vector Machine) fue utilizado por primera vez en la clasificación de texto en 1998 por T. Joachims [12]. En términos geométricos, se puede ver como el intento de encontrar un espacio n-dimensional, que permita separar los ejemplos de entrenamiento positivos de los negativos, permitiendo especificar el margen más amplio posible. El objetivo

de este algoritmo es encontrar el hiperplano óptimo que maximice la distancia entre los casos positivos y los casos negativos [12].

Como argumenta Joachims [12], las máquinas de vectores de soporte ofrecen dos grandes ventajas para la categorización de texto:

• Evitan los problemas de sobrecarga de pruebas en espacios de grandes dimensiones.
• Realizan una optimización global, sin óptimos locales.

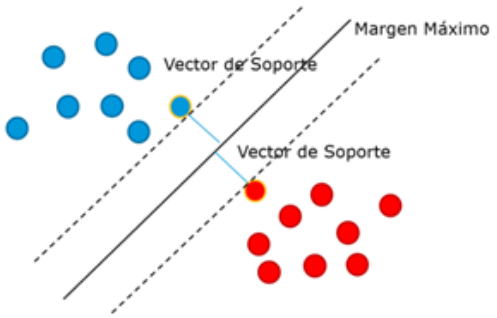

Figura 8: Modelo de máquinas de vectores de soporte lineal.

3.4.2 Desarrollo de los algoritmos de clasificación

Teniendo en cuenta la base teórica de los modelos se procedió a implementar los algoritmos de aprendizaje automático.

-Algoritmo de regresión logística:

Para la implementación del algoritmo, se utilizó la función *LogisticRegression* de la librería *sklearn.linear-model* [13], esta función proporciona diferentes hiper-parámetros que permitieron el ajuste del modelo de acuerdo a nuestros datos.

El algoritmo de regresión logística se compone de tres pasos:

1. Estimación de los valores TF-IDF obtenidos de la creación del modelo. El algoritmo trabajará con el concepto de puntos, el cual consiste en asignar a cada segmento un puntaje (y).

2. Considerando la ecuación:

$$y = a + bx_1 + cx_2 + \cdots + dx_n \quad (3)$$

$$a, b, c, d = \textit{Coeficientes} \quad (4)$$

$$n = \textit{Longitud del los datos} \quad (5)$$

$$x_1, x_2 ... x_n = \textit{V. Independiente (valores TF-IDF)} \quad (6)$$

$$y = \textit{V. Dependiente (clases)} \quad (7)$$

Se encuentran los valores óptimos para los coeficientes y se estima el valor de la predicción: si $y >= 0.5$ el párrafo se clasificará como positivo y si $y < 0.5$ el párrafo se clasificará como negativo.

3. Para algunos valores de las variables dependientes, el valor de y puede ser > 1 ó < 0. Por esta razón, se realiza una restricción, permitiendo que y tenga solo valores del rango 0 y 1.

$$\textit{Para } y > 0, y = e^{a+bx_1+cx_1+\cdots+dx_n}. \quad (8)$$

$$\textit{Para } y < 1, y = \frac{e^{a+bx_1+cx_1+\cdots+dx_n}}{e^{a+bx_1+cx_1+\cdots+dx_n}+1}. \quad (9)$$

-Algoritmo de Máquinas de vectores de soporte (SVM):

Para la implementación del algoritmo, se utilizó la función *SVM* de la librería *sklearn.svm* [14], esta función proporciona diferentes hiper-parámetros que permitieron el ajuste del modelo de acuerdo a los datos obtenidos.

Los siguientes son los tres pasos del algoritmo SVM:
1. Se estiman los valores TF-IDF obtenidos de la creación del modelo. Utilizando el kernel lineal, se cuantifica la similitud de un par de observaciones usando la correlación de Pearson.

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}. \quad (10)$$

2. Se encuentran los valores óptimos para los coeficientes y se estima el valor de la predicción.

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} > 0, \ \textit{si } y_i = 0. \quad (11)$$

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} > 0, \ \textit{si } y_i = 1. \quad (12)$$

$$f(x^*) = \beta_0 + \beta_{1x_1^*} + \cdots + \beta_{px_p^*}. \quad (13)$$

$$\beta_0 + \beta_1 + \cdots + \beta_p = \textit{Coeficientes del hiperplano.} \quad (14)$$

Si la ecuación (13) es positiva, se asigna a la clase 1, y si es negativa, a la clase 0. Esto quiere decir que clasifica cada nueva observación en función de a qué

lado de hiperplano pertenezca, es decir, en función del signo de la ecuación (13); y al igual que con el *maximal margin classifier*, solo las observaciones que se encuentran sobre el margen o que lo violan (vectores de soporte) afectarán al hiperplano y, por lo tanto, al clasificador obtenido.

3. Al proceso de optimización del hiperplano se incorpora un parámetro de regularización C, el cual controla la severidad permitida de las violaciones de las n observaciones sobre el margen e hiperplano, y a la vez, el equilibrio sesgo-varianza.

3.6 Validación de los algoritmos de clasificación:

Con el propósito de comparar y seleccionar el algoritmo de clasificación más eficiente, se utilizó la validación *k-fold cross validation*. La validación cruzada de k iteraciones consiste en dividir los datos originales en k subconjuntos y, al momento de realizar el entrenamiento, se va a tomar cada k subconjunto como conjunto de prueba del modelo, mientras que el resto de subconjuntos (k-1) se tomará como conjunto de entrenamiento. El proceso de validación cruzada se repetirá k veces, y en cada iteración se tomará un conjunto de prueba diferente, siendo el resto de datos el conjunto de entrenamiento. Al finalizar las k iteraciones, se calcula el promedio de los resultados de precisión y error obtenidos para cada subconjunto de prueba [15]. En general, se recomienda una validación cruzada estratificada de 10-fold para estimar la precisión, debido a su sesgo y varianza relativamente bajos [16].

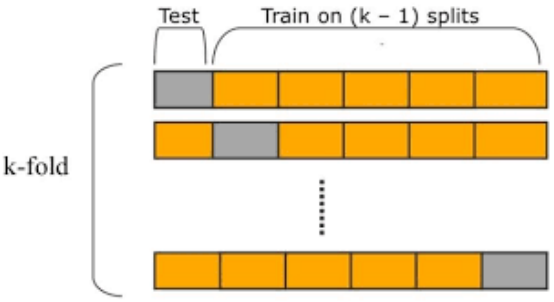

Figura 9: Proceso de funcionamiento de la validación k-fold.

Para la validación de los algoritmos se utilizan cuatro medidas de desempeño: accuracy, error rate, precision y recall:

Accuracy:Fracción de todas las instancias en la que la predicción del clasificador es correcta (para la clase positiva o negativa) [17].
Error Rate:Fracción de todas las instancias en la que la predicción del clasificador es incorrecta (para la clase positiva o negativa) [17].
Precision:Fracción de las predicciones positivas que son correctas [17].
Recall:Fracción de las instancias positivas que el clasificador identifica correctamente como positiva [17].
***F1-Score*:** Combina precisión y recall en un solo valor, con la finalidad de poder comparar el rendimiento general de los diferentes modelos que se utilicen [17].

3.6.1 Resultados obtenidos:

Para cada algoritmo y para cada valor de k se obtuvieron las medias de desempeño. Estos resultados se compararán a continuación:

Tabla IV: Validación k-fold - clasificador de *efficiency*.

	Medidas \ K	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	Media
	Accuracy	85 %	95 %	95 %	89 %	84 %	84 %	89 %	89 %	95 %	84 %	89 %
	Precision	90 %	90 %	89 %	88 %	82 %	63 %	100 %	89 %	90 %	73 %	85 %
RL	Recall	83 %	100 %	100 %	88 %	90 %	100 %	82 %	89 %	100 %	100 %	93 %
	F1-Score	87 %	95 %	94 %	88 %	86 %	77 %	90 %	89 %	95 %	84 %	88 %
	Error Rate	15 %	5 %	5 %	11 %	16 %	16 %	11 %	11 %	5 %	16 %	11 %
	Accuracy	90 %	95 %	89 %	89 %	79 %	84 %	84 %	79 %	95 %	84 %	87 %
	Precision	92 %	90 %	80 %	80 %	75 %	63 %	90 %	78 %	91 %	73 %	81 %
SVM	Recall	92 %	100 %	100 %	100 %	90 %	100 %	82 %	78 %	100 %	100 %	94 %
	F1-Score	92 %	95 %	89 %	89 %	82 %	77 %	86 %	78 %	95 %	84 %	87 %
	Error Rate	10 %	5 %	11 %	11 %	21 %	16 %	16 %	21 %	5 %	16 %	13 %

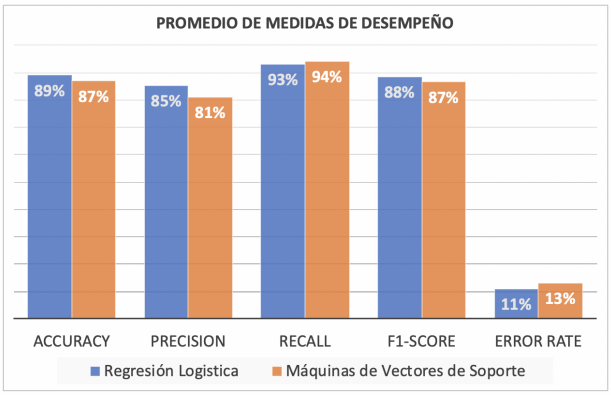

Figura 10: Gráfico de comparación del promedio de las medidas de desempeño - Variable *efficiency*.

Como podemos observar en la Tabla IV de la variable *efficiency*, se obtuvieron en los dos algoritmos el mejor resultado en K=2 alcanzando 94 % de *Accuracy*, 6 % de *Error Rate*, *Precision* de 100 %, *Recall* de 90 % y *F1-Score* de 95 %.

Se destaca que el algoritmo de regresión logística obtuvo los mejores resultados promedios de las medidas desempeño, sobresaliendo un *Error Rate* de 11 %, *Precision* de 85 %, *Accuracy* de 89 % y desde un punto de vista global (es decir, utilizando la métrica F1 con un valor de 84 %), fue el algoritmo que presentó el mejor comportamiento al momento de la clasificación, como se observa en la Fig. 10.

Tabla V: Validación k-fold - clasificador de *band gap*

	Métricas\K	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	Media
	Accuracy	88 %	81 %	94 %	81 %	81 %	81 %	75 %	75 %	88 %	94 %	84 %
	Precision	92 %	82 %	100 %	77 %	85 %	77 %	78 %	67 %	85 %	92 %	83 %
RL	Recall	92 %	90 %	90 %	100 %	92 %	78 %	100 %	100 %	100 %	100 %	94 %
	F1-Score	92 %	86 %	95 %	87 %	88 %	87 %	78 %	80 %	92 %	96 %	88 %
	Error Rate	12 %	19 %	6 %	19 %	19 %	19 %	25 %	25 %	13 %	6 %	16 %
	Accuracy	88 %	81 %	81 %	81 %	88 %	81 %	75 %	69 %	88 %	81 %	81 %
	Precision	92 %	82 %	83 %	77 %	86 %	77 %	69 %	62 %	85 %	83 %	80 %
SVM	Recall	92 %	90 %	91 %	100 %	100 %	100 %	100 %	100 %	100 %	91 %	96 %
	F1-Score	92 %	86 %	87 %	87 %	92 %	87 %	82 %	76 %	92 %	87 %	87 %
	Error Rate	12 %	19 %	19 %	19 %	13 %	19 %	25 %	31 %	13 %	19 %	19 %

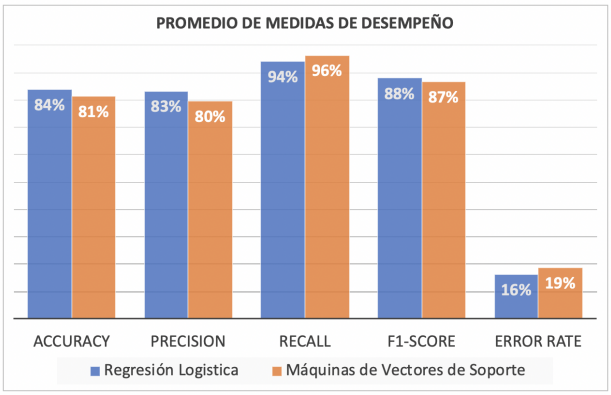

Figura 11: Gráfico de comparación del promedio de las medidas de desempeño - variable *band gap*

Como podemos observar en la Tabla V de la variable *band gap*, el mejor resultado se obtuvo con el algoritmo de regresión logística en K=3 alcanzando un 94 % de *Accuracy*, *Error Rate* de 6 %, *Precision* de 100 %, *Recall* de 90 % y *F1-Score* de 95 %.

Se destaca que el algoritmo de regresión logística obtuvo los mejores resultados promedios de las medidas desempeño, sobresaliendo *Error Rate* de 16 %, *Precision* de 83 %, *Accuracy* de 84 % y desde un punto de vista global (es decir, utilizando la métrica F1 con un valor de 88 %), fue el algoritmo que presentó el mejor comportamiento al momento de la clasificación, como se observa en la Fig. 11.

Tabla VI: Validación k-fold - clasificador *synthesis method*

	Métricas	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	Media
	Accuracy	72 %	72 %	94 %	94 %	89 %	88 %	76 %	82 %	88 %	94 %	85 %
	Precision	75 %	60 %	100 %	100 %	89 %	100 %	60 %	88 %	82 %	86 %	84 %
RL	Recall	67 %	86 %	90 %	89 %	89 %	60 %	100 %	78 %	100 %	100 %	86 %
	F1-Score	71 %	71 %	95 %	94 %	89 %	75 %	75 %	83 %	90 %	93 %	84 %
	Error Rate	28 %	28 %	6 %	6 %	11 %	12 %	24 %	18 %	12 %	6 %	15 %
	Accuracy	78 %	71 %	94 %	89 %	94 %	94 %	76 %	76 %	88 %	94 %	86 %
	Precision	86 %	60 %	100 %	100 %	100 %	63 %	86 %	89 %	100 %	100 %	88 %
SVM	Recall	67 %	86 %	90 %	78 %	89 %	80 %	83 %	67 %	89 %	83 %	81 %
	F1-Score	75 %	71 %	95 %	88 %	94 %	89 %	71 %	75 %	89 %	91 %	84 %
	Error Rate	22 %	29 %	6 %	11 %	6 %	6 %	24 %	24 %	12 %	6 %	14 %

Como podemos observar en la Tabla VI de la variable *synthesis method*, se obtuvieron en los dos algoritmos el mejor resultado en K=3 alcanzando un 94 % de *Accuracy*, 6 % de *Error Rate*, *Precision* de 100 %, *Recall* de 90 % y *F1-Score* de 95 %.

Se destaca que tanto el algoritmo de regresión logística, como el algoritmo de máquinas de vectores de soporte

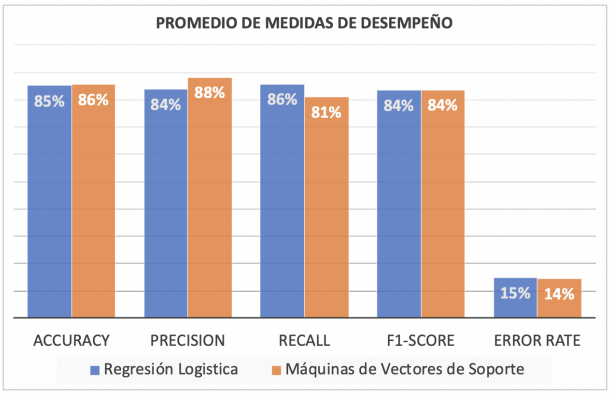

Figura 12: Gráfico de comparación del promedio de las medidas de desempeño - variable *method sythesis*

obtuvieron resultados promedios muy similares de las medidas desempeño, con *Error Rate* de 15 %-14 %, *Precision* de 84 % - 88 %, *Accuracy* de 85 % - 86 % y *F1-Scrore* de 84 % - 84 %, como se observa en la Fig 12. Esto nos indica que cualquiera de los algoritmos tiene el mismo o similar desempeño en la clasificación.

Tabla VII: Validación k-fold - clasificador de *annealing*

	Métricas	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	Media
	Accuracy	89 %	67 %	78 %	78 %	67 %	56 %	78 %	63 %	63 %	63 %	70 %
	Precision	86 %	60 %	100 %	83 %	67 %	57 %	67 %	63 %	80 %	80 %	74 %
RL	Recall	100 %	75 %	71 %	83 %	80 %	80 %	100 %	100 %	67 %	67 %	82 %
	F1-Score	92 %	67 %	83 %	83 %	73 %	67 %	80 %	77 %	73 %	73 %	77 %
	Error Rate	11 %	33 %	22 %	22 %	33 %	44 %	22 %	38 %	38 %	38 %	30 %
	Accuracy	89 %	44 %	78 %	67 %	78 %	56 %	67 %	63 %	63 %	38 %	64 %
	Precision	86 %	43 %	86 %	67 %	71 %	56 %	57 %	62 %	80 %	60 %	67 %
SVM	Recall	100 %	75 %	86 %	100 %	100 %	100 %	100 %	67 %	50 %	100 %	88 %
	F1-Score	92 %	55 %	86 %	80 %	83 %	71 %	73 %	77 %	73 %	55 %	74 %
	Error Rate	11 %	56 %	22 %	33 %	22 %	44 %	33 %	38 %	38 %	62 %	36 %

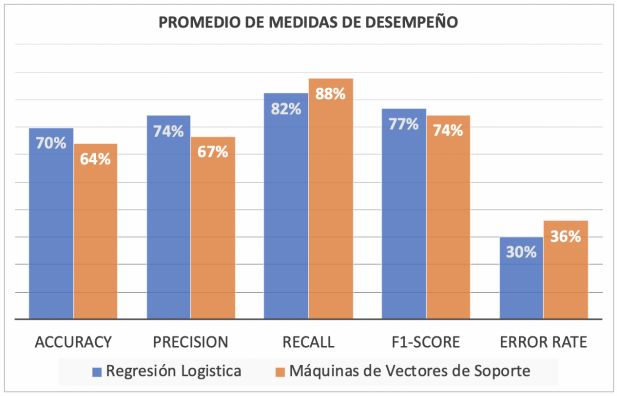

Figura 13: Gráfico de comparación del promedio de las medidas de desempeño - variable *annealing*

Como podemos observar en la Tabla VII de la variable *annealing*, el mejor resultado se obtuvo con el algoritmo de regresión logística en K=1 alcanzando 89 % de *Accuracy*, 11 % de *Error Rate*, *Precision* de 86 %, *Recall* de 100 % y *F1-Score* de 92 %.

Se destaca que el algoritmo de regresión logística obtuvo los mejores resultados promedios de las medidas desempeño, sobresaliendo *Error Rate* de 30 %, *Precision* de 74 %, *Accuracy* de 70 % y desde un punto de vista global (es decir, utilizando la métrica F1 con un valor de 77 %), fue el algoritmo que presentó el mejor comportamiento al momento de la clasificación, como se observa en la Fig 13.

Tabla VIII: Validación k-fold - clasificador de *coefficient absorption*

	Métricas\K	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	Promedio
	Accuracy	78 %	88 %	98 %	89 %	100 %	88 %	88 %	88 %	88 %	88 %	90 %
	Precision	98 %	92 %	96 %	96 %	98 %	50 %	80 %	100 %	100 %	98 %	92 %
RL	Recall	60 %	100 %	89 %	83 %	100 %	100 %	100 %	80 %	75 %	80 %	88 %
	F1-Score	75 %	88 %	94 %	91 %	100 %	67 %	89 %	89 %	86 %	89 %	88 %
	Error Rate	22 %	12 %	2 %	11 %	0 %	13 %	13 %	13 %	13 %	13 %	10 %
	Accuracy	78 %	100 %	100 %	89 %	89 %	88 %	88 %	88 %	88 %	88 %	89 %
	Precision	98 %	99 %	100 %	97 %	75 %	50 %	80 %	100 %	99 %	95 %	89 %
SVM	Recall	60 %	100 %	100 %	83 %	100 %	100 %	100 %	80 %	75 %	80 %	88 %
	F1-Score	75 %	100 %	100 %	91 %	86 %	67 %	89 %	89 %	86 %	89 %	87 %
	Error Rate	22 %	0 %	0 %	11 %	11 %	13 %	13 %	13 %	13 %	13 %	11 %

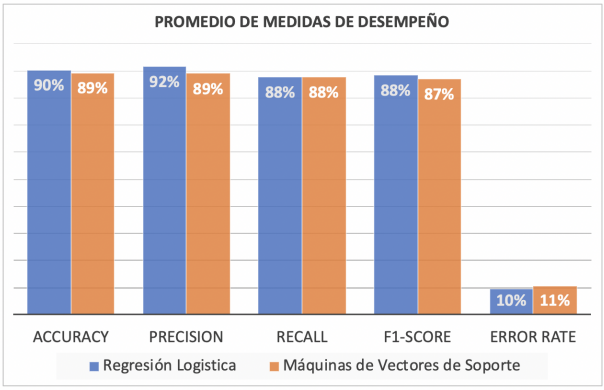

Figura 14: Gráfico de comparación del promedio de las medidas de desempeño - Variable *coefficient absorption*

Como podemos observar en la Tabla VIII de la variable *coefficient absorption*, el mejor resultado se obtuvo con el algoritmo de regresión logística en K=5 alcanzando 100 % de *Accuracy*, 10 % de *Error Rate*, *Precision* de 98 %, *Recall* de 100 % y *F1-Score* de 100 %.

Se destaca que tanto el algoritmo de regresión logística, como el algoritmo de máquinas de vectores de soporte obtuvieron resultados promedios muy similares de las medidas desempeño, sobresaliendo *Error Rate* de 10 % - 11 %, *Precision* de 92 % - 89 %, *Accuracy* de 90 %-89 % y *F1-Scrore* de 88 %- 87 %, como se observa en la Fig. 14. Esto nos indica que cualquiera de los algoritmos tiene el mismo o similar desempeño a la hora de la clasificación.

Después de obtener los resultados de cada validación realizada a cada uno de los clasificadores de las variables de interés y de un análisis completo de estos, se concluye que el algoritmo de Regresión Logística es el mejor a implementar en el sistema de reconocimiento de entidades, debido a que mostró los mejores resultados en cada validación de cada variable, por medio de diferentes medidas de desempeño que lo sustentaron.

IV-D. Representación

El proceso de representación se encargará de exportar los datos extraídos y clasificados de cada uno de los artículos a analizar por medio de una hoja de datos de tipo **.xlsx**. Estos datos están compuestos por: los párrafos positivos obtenidos a partir del proceso de clasificación y la información de identificación de cada articulo (titulo y año de publicación), esta extracción se realizó por medio de las herramienta de NLTK.

Figura 15: Diagrama de bloques del proceso de representación

V. RESULTADOS DEL SISTEMA

El resultado final del sistema de reconocimiento de entidades permite al usuario conocer de manera rápida y con una precisión considerable las variables de interés: *Efficiency, Band Gap, Synthesis Method, Annealing y Absorption Coefficient* obtenidos de la parte experimental de cada documento.

Estos resúmenes están representados en una hoja de datos de Excel, la cual se encuentra organizada en columnas que representan los resultados obtenidos por la extracción y clasificación del documento además de su fecha de publicación. Los resultados contienen el término clave de la Tabla I de cada variable de interés, acompañado de un párrafo explicativo, el cual tiene como propósito brindarle contexto al usuario del término clave de la variable.

VI. CONCLUSIÓN

Se desarrolló un sistema automático de reconocimiento de entidades como herramienta para extraer, clasificar y representar parámetros de acuerdo a las variables de interés. *Efficiency, Band Gap, Synthesis Method, Annealing y Absorption Coefficient*. Este sistema de clasificación segmenta la totalidad del texto y agrupa en diccionarios a partir de la presencia de parámetros de interés dentro del segmento como lo son las unidades de interés y valores numéricos asociados a una variable. De esta manera se logra favorecer a los investigadores a realizar sus investigaciones de una manera más eficiente, con el propósito de mejorar el desempeño de módulos fotovoltaicos y lograr una mejor aplicación con un menor costo.

La librería NLTK de Python resultó ser una herramienta ideal para los procesos de segmentación, análisis de lenguaje y agrupación de segmentos de acuerdo a los parámetros de interés; luego de la extracción de los artículos científicos provenientes de diferentes revistas científicas, se clasificó manualmente de acuerdo a los parámetros de interés *Efficiency, Band Gap, Synthesis Method, Annealing y Absorption Coefficient* con el fin de crear una nueva que permita categorizar los datos que se quieren clasificar en el sistema. De esta manera se permitió definir el tipo de aprendizaje automático que se requiere para el proceso de clasificación del sistema eligiendo los algoritmos más propicios para la predicción de clases binarias.

Para realizar el proceso de clasificación del sistema automático de reconocimiento de entidades fueron usados dos métodos, cada uno con su correspondiente algoritmo: regresión logística y máquinas de vectores de soporte. Estos algoritmos se implementaron con el propósito de comparar y determinar cuál de los dos tenía mejor desempeño y menor tasa de error al momento de predecir los segmentos positivos y negativos del sistema.

Con la técnica de validación cruzada k-fold se obtiene un bajo error en la evaluación del rendimiento de los algoritmos de clasificación. De los resultados obtenidos a partir de esta validación, se determinó que el algoritmo de regresión logística consigue los resultados más aceptables en la detección de los segmentos positivos (la información relevante de los parámetros) para cada clasificador de las variables de interés.

VII. TRABAJO A FUTURO

Existe la posibilidad de modificar el código de manera que a medida que se ingresan documentos a analizar y estos superen el umbral de aceptación definido, este último sea agregado de manera automática al código, sirviendo como referencia para los documentos que sean ingresados en ejecuciones futuras.

De cara a futuros trabajos se recomienda aumentar la base de datos para el entrenamiento del aprendizaje automático. Cuanto mayor sea el número segmentos tanto positivos como negativos de la base de datos, mejor desempeño tendrá el sistema a la hora de realizar la clasificación de los párrafos positivos, identificando con mayor precisión la información de mayor relevancia de cada variable de interés.

La herramienta de extracción desarrollada para este artículo puede modificarse de manera que funcione para otro tipo de materiales semiconductores diferentes a la kesterita, esto se logra cambiando las variables deseadas y la base de datos de los párrafos de interés con los que se realiza el entrenamiento del aprendizaje automático. De esta manera es posible generar otra herramienta, que siguiendo la misma estructura y procedimiento, pueda extraer de manera automática información contenida de cualquier documento en texto plano (**.txt**).

REFERENCIAS

[1] N. P. A. Arredondo, "Método Semisupervisado para la Clasificación Automática de Textos de Opinión," *Instituto Nacional de Astrofísica, Óptica y Electrónica*, pp. 10,80, 2009.

[2] E. Huerta Mascotte, R. I. Mata Chávez, J. M. Estudillo Ayala, J. M. Sierra Hernández, I. Guryev, and R. A. Lizárraga Morales, "Solar cell characteristics study for solar energy efficient use," *Acta Universitaria*, vol. 26, no. NE-1, pp. 30–34, 2016.

[3] W. A. Chamorro Coral and S. Urrego Riveros, "Celdas solares orgánicas, una perspectiva hacia el futuro," *Elementos*, vol. 2, no. 2, 2013.

[4] J. Oyola and G. Gordillo, "Estado del arte de los materiales fotovoltaicos y de la tecnología solar fotovoltaica," *Prospectiva*, vol. 5, no. 2, pp. 11–15, 2007.

[5] K. Pal, P. Singh, A. Bhaduri, and K. B. Thapa, "Current challenges and future prospects for a highly efficient (¿20 %) kesterite CZTS solar cell: A review," *Solar Energy Materials and Solar Cells*, vol. 196, no. July 2018, pp. 138–156, 2019.

[6] V. Karade, A. Lokhande, P. Babar, M. G. Gang, M. Suryawanshi, P. Patil, and J. H. Kim, "Insights into kesterite's back contact interface: A status review," *Solar Energy Materials and Solar Cells*, vol. 200, no. April, p. 109911, 2019.

[7] "sklearn.datasets.load_files — scikit-learn 0.22.2 documentation."

[8] Zhixiang, Xu, M. Chen, K. Q. Weinberger, and F. Sha, "An alternative text representation to TF-IDF and Bag-of-Words," no. January, 2013.

[9] "Técnicas avanzadas de recuperación de información: Frecuencias y pesos de los términos de un documento."

[10] M. Lizares Castillo, "Comparación de modelos de clasificación: regresión logística y árboles de clasificación para evaluar el rendimiento académico," *Universidad Nacional Mayor de San Marcos*, 2017.

[11] P. N. Kallus, C. Tech, S. B. Hwang, B. Yellin, H. Zhang, K. Lotay, O. Winarunke, and S. Phadke, "CS 5785 – Applied Machine Learning – Lec . 5 Recap - Logistic Regression," no. 2, pp. 1–11, 2017.

[12] L. Kirkwood and L. Kirkwood, "Chimerica," *Chimerica*, 2015.

[13] "sklearn.linear_model.LogisticRegression — scikit-learn 0.22.2 documentation."

[14] "sklearn.svm.SVC — scikit-learn 0.22.2 documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[15] "Introducción a la Validación Cruzada (k-fold Cross Validation) en R."

[16] S. Agarwal, *Data mining: Data mining concepts and techniques*, 2014.

[17] Dra. Helena Montserrat Gómez Adorno, "Introducción al aprendizaje automático."