Facultad de Ciencias Naturales y Exactas

Departamento de Ciencia de la Computación

Trabajo de Diploma en Opción al Título de Licenciado en Ciencia de la Computación

Título:
Aprendizaje Profundo para el Perfilado de Usuarios en Redes Sociales

Autor: Roberto Labadie Tamayo

Tutores: Dr. Daniel Castro Castro

M.Sc. Reynier Ortega Bueno

Curso 2020-2021

Contenido

Lista de Figuras

Lista de Tablas

Introducción

Internet se define como la interconexión mundial de redes individuales operadas por el gobierno, la industria, el mundo académico y las partes privadas. En cuestión de muy pocos años, Internet se consolidó como una plataforma muy poderosa que ha cambiado para siempre la forma de comunicarse. Esta manera tan simple y accesible de compartir datos, ha propiciado un desplazamiento de los entes sociales hacia el uso casi exclusivo de este medio.

Un rol fundamental dentro de este universo comunicativo lo juegan las redes sociales donde de acuerdo al sitio DataReportal[1] a finales de 2020 existían más de 3.8 mil millones de usuarios de los 4.5 mil millones de personas conectadas a Internet, lo cual implica un crecimiento desmedido de la cantidad de datos multimodales que se genera a diario.

El mundo del Big Data (Riahi and Riahi, 2018) en el cual nos sumerge este hecho, reporta una situación beneficiosa para el desarrollo de procesos sociales, en cuanto a la cantidad de información brindada por los usuarios . Estos procesos van desde estudios de marketing donde la retroalimentación a partir de opiniones permiten dirigir de manera efectiva la promoción hacia grupos sociales específicos, hasta aplicaciones en el área de la Interacción Humano-Computadora (*Human-Computer Interaction* HCI), donde el conocimiento de las características físicas y psicológicas de las personas permite personalizar la interfaz de comunicación. Sin embargo un cúmulo de información tan significativo, se hace imposible de tratar en un espacio de tiempo razonable a menos que se haga de manera automatizada.

Diversos estudios se han dirigido al diseño de métodos para manejar tal cantidad de información y realizar inferencias a partir de la misma, a la vez que han contribuido al desarrollo en ramas tales como el Procesamiento del Lenguaje Natural (*Natural Langauage Processing* NLP) y Visión por Computadoras (*Computer Vision* CV) en el área de la Inteligencia Artificial (*Artificial Intelligence* AI).

Dentro del NLP, la tarea de Perfilado de Autores (*Author Profiling AP*) (Rosso et al., 2019; Rosso and Rangel Pardo, 2020) se encarga específicamente de realizar un análisis de la información textual elaborada por una persona que permita establecer atributos y patrones de comportamiento para caracterizarla en cuanto a sexo, rango de edades o rasgos personales (e.g si la persona es extrovertida o no, ideología política). Sin embargo, para el AP el hecho de que la información recuperada de estos textos varíe enormemente en términos de su formato aun cuando proviene de la misma persona, sumado a que las secuencias texuales constituyen información digital no estructurada, hacen desafiante el proceso de analizarla y clasificarla automáticamente.

Inicialmente este tipo de tareas se desarrollaban sobre contenido generado en textos formales, periódicos, cartas o revistas, sin embargo determinar el perfil de una persona mediante el análisis de su cuenta en una red social ha tomado un gran auge en los últimos años (Rangel et al.,

[1] https://datareportal.com/

2018a; Rangel and Rosso, 2019; Cimino et al., 2020). Las redes sociales además de haber logrado acelerar la comunicación entre las personas así como reunir varias formas del pensamiento individual en un mismo espacio, se han convertido en un medio donde se desarrollan procesos negativos tales como, la divulgación de discursos de odio (Rangel et al., 2021), *bullying* o de noticias falsas (Rangel et al., 2020a). Este hecho introduce nuevos retos al AP con tareas que involucran la detección de elementos altamente subjetivas como la ofensa, la toxicidad en el lenguaje y otros patrones psicológicos de comunicación que hacen más complejo el perfilado en relación a otras tareas.

La mayor parte de los avances en el desarrollo de sistemas de AP han sentado sus bases dentro del ambiente académico, reuniendo estudios tanto de la Ciencia de la Computación como de la Lingüística. Algunas de las campañas de evaluación más importantes donde se han compartido tareas de este tipo son *Plagiarism, Authorship and Social Software Misuse* PAN[2] y *Evaluation of NLP and Speech Tools for Italian* EVALITA[3], los que se han dirigido últimamente al análisis del genero textual de micro-blogging con un enfoque multilingüe, empleado en medios sociales como Twitter[4].

Tradicionalmente se han empleado dos tipos de acercamientos que han probado empíricamente ser efectivos para tratar el AP en redes sociales: los *basados en estilo* y los *basados en contenido*. Las propuestas basadas en estilo se refieren al hecho de analizar como los autores se expresan cuando escriben, en cambio la basada en contenido se apoyan en el área temática del texto analizado. La mayor contribución de varios trabajos se ha basado en la selección de atributos que permitan medir el estilo autor y el contenido simultáneamente mediante el empleo de métodos de Aprendizaje de Máquina (*Machine Learning* ML).

Un proceso crítico en este tipo de propuestas es la selección de rasgos para representar vectorialmente los elementos del texto, en el cual pueden ser eliminados elementos que ayuden al modelo a discernir entre una clase u otra, en el caso de tareas de clasificación, pero también se puede introducir información ruidosa y de igual forma afectar su desempeño.

En los últimos años dentro del Machine Learning ha tenido un auge enorme el Aprendizaje Profundo (*Deep Learning* DL) en las áreas de CV y NLP, estableciendo nuevos estados del arte en la mayoría de sus tareas (Alom et al., 2019). Este auge estuvo condicionado por las capacidades de procesamiento de las nuevas computadoras y la cantidad de datos disponibles. Los modelos de DL han mostrado una gran habilidad para aprender representaciones de rasgos (*features*) con un alto nivel de abstracción. Estas representaciones capturan elementos que son posiblemente omitidos mediante la extracción manual de features y que facilitan el proceso de inferencia de los propios modelos de DL y de los métodos más tradicionales de ML.

Dentro del AP también se ha introducido el Aprendizaje Profundo con resultados alentadores. Sin embargo la mayoría de los modelos alcanzan mejores resultados al analizar la tarea a la que se orienta el perfilado cuando se emplean para clasificar mensajes individuales en lugar del perfil completo, e.g., se desempeñan mejor en la tarea de detección de odio en un tweet que en la detección de un perfil que tiende a usar un discurso de odio.

Por otro lado, las necesidades que cubren las tareas de AP incluyen la no sensibilidad de los sistemas ante la variación del idioma en el que la persona postee un mensaje o escriba una carta, es por ello que urge la extensión de los modelos de AI propuestos al esquema multilingüe

[2]https://pan.webis.de/
[3]http://www.evalita.it/
[4]https://twitter.com/

que predomina en redes sociales como Twitter.
Este trabajo de tesis se enmarca en el perfilado de autores en redes sociales empleando técnicas de Aprendizaje Profundo para el modelado y la clasificación de los perfiles teniendo en cuenta el enfoque multilingüe propio de este medio.

Problemática

La mayoría de los trabajos existentes que emplean DL para resolver tareas de perfilado de autores en redes sociales, tanto tradicionales (e.g., determinar rango de edades, sexo, etc.) como las más recientes (e.g. detección de divulgadores de discursos de odio en redes sociales o noticias falsas) ven afectado su desempeño al tratar con las secuencias muy largas que se pueden generar al analizar un perfil completo a la vez, en vez de mensaje a mensaje. La perdida de información (*information vanishing*) en este tipo de secuencias en términos de relaciones a largo plazo que presentan las estructuras sintácticas del lenguaje, es la causa fundamental de esta dificultad. Por esta razón sería conveniente dividir el problema de "clasificar un perfil" en subproblemas y construir una arquitectura modular, en la que primero se modele el perfil y luego este sea clasificado.
Por otro lado en Cuba existe un limitado tratamiento y aplicación de este tipo de métodos automatizados para llevar acabo tareas de impacto como las mencionadas, ya sea en el área forense o empresarial .

Hipótesis

Dividir el proceso de clasificación de un perfil dentro de una tarea en: (1) codificar los mensajes individualmente y (2) modelar-clasificar el perfil, puede ayudar a prevenir la perdida de información generada del análisis de largas secuencias. Esto sumado a la capacidad de representar la información no estructurada de los métodos de DL puede influir positivamente en la precisión de las predicciones. Además, proponer un modelo lo suficientemente robusto, podría incentivar el empleo de esos métodos automatizados de AP en nuestro país.

Objetivos

Objetivo General

Diseñar una arquitectura modular basada en Aprendizaje Profundo para resolver la tarea de Perfilado de Autores en redes sociales teniendo en cuenta un enfoque multilingüe.

Objetivos Específicos

1. Diseñar una arquitectura que capture rasgos abstractos que permitan clasificar un perfil de usuario de una red social atendiendo tanto a tareas relacionadas con características demográficas de los autores, como con aspectos psicológicos de los mismos.

2. Extender la arquitectura propuesta a un enfoque multilingüe de las tareas evaluadas.

3. Evaluar y analizar el empleo de arquitecturas modulares basadas en DL sobre colecciones de datos propuestas en tareas de AP compartidas en la plataforma PAN (2019, 2020, 2021).

Estructura del Trabajo

Este trabajo esta estructurado en tres capítulos además del introductorio y una última sección en la que se describen las conclusiones de las modelaciones y se proponen caminos a seguir para trabajos futuros. Cada uno de los capítulos se listan a continuación:

- En el Capítulo 1 se describen de manera breve métodos del estado del arte así como conceptos básicos relacionados con el Machine Learning necesarios para la comprensión de este trabajo.
- El Capítulo 2 describe las tareas de AP en las que se evaluarán las arquitecturas propuestas, así como los datos anotados sobre los cuales se basan los procesos de entrenamiento y evaluación.
- En el Capítulo 3 se exponen los experimentos realizados en el proceso de ajuste de cada unos de los modelos y sus módulos sobre los idiomas estudiados.

.

Capítulo 1

Fundamentos

1.1. Estado del Arte

El Procesamiento del Lenguaje Natural, como cualquier tarea llevada a cabo por un algoritmo de Machine Learning, se basa en el análisis de un conjunto de rasgos del objeto a procesar, con la finalidad de realizar inferencias que permitan al sistema computarizado interactuar con el usuario.
Para la tarea de AP no es diferente, por lo que la mayoría de los trabajos se dirigen al desarrollo de una efectiva selección de rasgos, estableciéndose dos categorías fundamentales; los rasgos de estilo y los de contenido.
El análisis de estilo se enfoca en la detección de rasgos estilísticos del autor que sean lo suficientemente invariantes a lo largo de los pasajes escritos, pero que varíen de un autor a otro. Ejemplos de estos podrían ser la longitud media de las oraciones o la cantidad esperada de símbolos de puntuación, emoticones, preposiciones o adverbios en un párrafo.
Por otra parte el análisis de contenido se enfoca en la información contextual siguiendo la misma estrategia del análisis de estilo, i.e., llevar información estadística de la presencia de determinado conjunto de palabras o estructuras gramaticales. Este tipo de rasgos incluye el uso de n-gramas de palabras y caracteres, *slang words*[1], Bolsas de Palabras (*Bag of Words* BoW) (Pizarro, 2019; Valencia-Valencia et al., 2019), palabras concluyentes (e.g., finalmente, para concluir, en conclusión, etc.) y lexicones de palabras. Dentro de este último método para categorizar palabras, el más empleado es el sistema *Linguistic Inquiry and WordCount LIWC* (Pennebaker et al., 2015) el cual contiene alrededor de 70 diccionarios de lexicones divididos en categorías como; Preocupaciones personales (*Personal Concerns*), Discurso informal (*Informal Speech*), Impulsos (*Drives*) y necesidades fundamentales (*Basics Needs*), etc.

Estos dos grupos de rasgos, de estilo y contenido, son ortogonales puesto que los rasgos que se tienen en cuenta para capturar estilo, son precisamente aquellos que son independientes del tópico, por lo cual se ha hecho habitual el uso de enfoques que los combinen a ambos. Por otro lado el empleo de elementos contextuales implica introducir en el proceso de clasificación un sesgo hacia una clase que este más representada por determinado tema, e.g., según (Schler et al., 2006) estos rasgos pueden facilitar la determinación del sexo del autor ya que por ejemplo los hombres mayormente tienden a hablar de política y noticias, mientras que las mujeres

[1] *slang* se refiere a un lenguaje muy informal empleado por un grupo particular de personas.

se muestran más interesadas por la moda, fiestas y prendas de vestir; sin embargo es posible encontrar una mujer que regularmente postee tweets relacionados con deportes o automóviles, luego este perfil sería potencialmente mal clasificado por un modelo de AP basado en rasgos de contenido.
La mayoría de los trabajos de Machine Learning han usado métodos de clasificación tradicionales como Regresión Logística (Logistic Regression LR) (Valencia-Valencia et al., 2019), Máquina de Vectores de Soporte (*Support Vector Machines* SVM) (Pizarro, 2019) y Bosques Aleatorios (*Random Forest* RF) (Johansson, 2019) combinando conjuntos de rasgos que responden a estas clasificaciones.

Con la introducción del Deep Learning en el NLP esta tendencia a la extracción manual de rasgos de tipo estadístico ha sido desplazada por el aprendizaje de rasgos abstractos que representan a las estructuras gramaticales atendiendo no solamente a su significado semántico como elementos aislados del texto, sino que tienen en cuenta además el contexto en el que son empleados, facilitando la comprensión y clasificación de los textos tanto a los propios modelos de DL como a los tradicionales de ML. Este tipo de enfoques se basan fundamentalmente en el uso de *embeddings* preentrenados con distintas estrategias (Joo and Hwang, 2019; López-Santillán et al., 2019), principalmente Word2Vec (Mikolov et al., 2013), Glove(Pennington et al., 2014) y Fasttext(Bojanowski et al., 2016).
Las Redes Neuronales Artificiales como mecanismos de aprendizaje y clasificación, por su parte han logrado un desempeño superior a los métodos tradicionales de ML en muchas tareas de NLP, teniendo en cuenta que aprenden a decidir que elemento del texto tomar como rasgo significativo a la hora de modelar los objetos. Esquemas especializados en análisis de secuencias, como las Redes Neuronales Recurrentes (RNN) (Dias and Paraboni, 2019; Bakhteev et al., 2020) y las arquitecturas *Transformers* (Transformadoras) (Iyer and Vosoughi, 2020; Baruah et al., 2020) se han empleado satisfactoriamente para el perfilado de autores en los últimos años, pero también se ha extendido el uso de arquitecturas diseñadas inicialmente para el tratamiento de otro tipo de información estructurada como las imágenes, tal es el caso de las Redes Neuronales Convolucionales (CNN) (Petrík and Chudá, 2019; López-Santillán et al., 2019).

En los últimos años dentro de PAN se han propuesto tareas de perfilado que van desde la predicción de sexo y variedad del idioma hasta la detección de perfiles manejados por bots y perfiles que tienden a difundir discursos de odio en el medio social. En la mayoría de estas tareas los trabajos con un mejor desempeño se han enmarcado en técnicas tradicionales de ML, tal es el caso de (Basile et al., 2017) en la tarea *Gender and Language Variety Identification in Twitter at* PAN 2017, quienes emplearon rasgos construidos a partir de una medida no estándar de frecuencia de términos sobre unigrama de palabras y n-gramas de caracteres para entrenar una SVM y (Martinc et al., 2017) que combinaron n-gramas de palabras, caracteres y Elementos del Discurso (*Part of Speech* POS), además de información de sentimientos relacionada con el uso de emojis y conteo de elongación de caracteres, con otros rasgos de estilo para modelar el perfil y entrenar un Regresor Logístico.
La tarea *Bots and Gender Profiling in Twitter at* PAN 2019[2] consistente en determinar si una cuenta de Twitter pertenece a un bot o a un humano y en el segundo caso, inferir el sexo; (Pizarro, 2019) obtuvo la mejor precisión con una SVM, combinando representaciones de los

[2] https://pan.webis.de/clef19/pan19-web/author-profiling.html

tweets mediante *tf-idf* de n-gramas de palabras y caracteres. De igual forma (Johansson, 2019) trató de dar solución a la tarea empleando ML con Random Forest y rasgos de estilo como logitud de los tweets, número de letras mayúsculas, URLs, menciones, cantidad de RTs, así como rasgos de contenido, específicamente ocurrencia de términos y etiquetas de POS.
Nuevamente en PAN 2020 en la tarea *Profiling Fake News Spreaders on Twitter*[3], para detectar divulgadores de noticias falsas, (Pizarro, 2020) se basó en la combinación de vectores de *tf-idf* de n-gramas de palabras y caracteres para representar los tweets y clasificar los perfiles con una SVM. Mientras el modelo propuesto por (Buda and Bolonyai, 2020) consistió en un Regresor Logístico que combina las predicciones de cinco submodelos: (i) n-gramas con Regresor Logístico,(ii) n-gramas con SVM, (iii) n-gramas con Random Forest, (iv) n-gramas con XGBoost y (v) XGBoost con features de estilo.
Diversos modelos de DL empleando las arquitecturas citadas anteriormente (e.g., RNN y CNN) han sido propuestos a lo largo de estas competiciones, aunque ninguno de ellos mostró una precisión superior a la de los métodos tradicionales de ML. Sin embargo, en la tarea *Profiling Hate Speech Spreaders on Twitter at* PAN 2021 [4], consistente en determinar cuando un perfil era divulgador de contenido relacionado con el odio a grupos sociales específicos, el modelo con mejor desempeño, propuesto por (Siino et al., 2021) empleó una Red Neuronal Convolucional sobre la representación de los perfiles con *embeddings* de palabras.

Uno de los mayores problemas con los enfoques predominantes es que los rasgos extraídos son muy dependientes del contexto lo cual puede crear una alta sensibilidad de los modelos ante datos que no correspondan a los corpus con los que han sido refinados sus parámetros o como ya hemos expuesto tengan un sesgo hacia alguna clase. Por ejemplo (Newman et al., 2008) expone que las mujeres tienden a usar emoticones con mayor regularidad que los hombres, lo contrario de lo concluido por (Schwartz et al., 2013). Esto nos sugiere que métodos tan rigurosamente refinados como lo son los dependientes de rasgos manualmente extraídos tienen una menor robustez ante arquitecturas *End-to-End* como las de Deep Learing.

1.2. Marco Teórico

En este epígrafe se realiza un acercamiento hacia los temas y arquitecturas de Machine Learning necesarios para la comprensión de los modelos propuestos en el trabajo.

1.2.1. LSTM: Long Short-Term Memory Neural Networks

Las LSTM son un tipo de Redes Neuronales Recurrentes, las cuales están especializadas en el análisis de datos secuenciales. Las RNNs tienen una unidad principal (la unidad recurrente) la cual explora la secuencia de datos de entrada de un elemento a la vez, ya sea de izquierda a derecha o viceversa. Al analizar un elemento para determinar su estado oculto, se comparte la información capturada en pasos anteriores del recorriendo. Esto es, sea h_{t-1} el último estado oculto computado, $x_t \in \Re^d$ el $t - esimo$ elemento de la secuencia de entrada y f una función de no linealidad. El estado oculto actual, se define como:

$$h_t = f(W_x x_t + W_h h_{t-1} + b_h) \tag{1.1}$$

[3] https://pan.webis.de/clef20/pan20-web/author-profiling.html
[4] https://pan.webis.de/clef21/pan21-web/author-profiling.html

Donde $W_x \in Re^{n_u \times d}$ y $W_h \in Re^{n_u \times n_u}$ son matrices de parámetros y $b_h \in R^{n_u}$ el término de sesgo (*bias*), con n_u el número de neuronas y d la dimensión de los vectores que representan a los elementos de la secuencia. De esta forma la arquitectura aprende a considerar la información que tiene determinada influencia sobre el elemento de la secuencia que se analiza en cada paso como se muestra en la Figura 1.1, lo que le otorga una especie de "memoria".

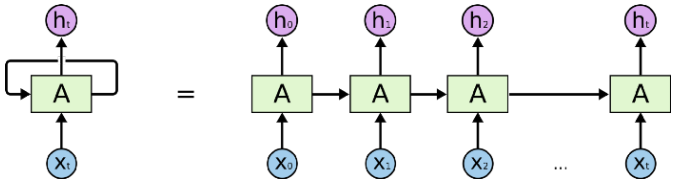


Figura 1.1: Red Neuronal Recurrente sobre la secuencia X_t. (Agarwala et al., 2017)

Sin embargo, debido a que en las RNNs durante el proceso de *backpropagation*, en el que se ajustan los parámetros de la red, cada neurona en la unidad principal calcula su gradiente para un paso del recorrido de la secuencia con respecto a su estado en el paso posterior, mediante la ley de la cadena, ocurre un decrecimiento exponencial de los valores de las derivadas parciales conocido como *gradient vanishing*, lo que hace que los parámetros a penas se actualicen y se dificulte el aprendizaje de relaciones a largo plazo, de aquí su "Short-Term Memory".
Esta limitación es lo que las LSTM tratan de solucionar introduciendo compuertas que deciden que información preservar u "olvidar" de los estados previos en el recorrido por la secuencia de la siguiente forma:
Sean $W_f, W_i, W_o \in \Re^{n_u \times d}$ y $U_f, U_i, U_o \in \Re^{n_u \times n_u}$ las matrices de parámetros de la compuerta de "olvidar ", entrada y salida respectivamente y $b_f, b_i, b_o \in \Re^{n_u}$ sus respectivos términos de bias:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \end{aligned} \tag{1.2}$$

Una codificación potencial $\hat{c}_t$ considerando el elemento x_t de la secuencia y el estado previo h_{t-1} esta dado por:

$$\hat{c}_t = \sigma(W_c x_t + U_c h_{t-1} + b_c) \tag{1.3}$$

Donde $W_c \in \Re^{n_u \times d}$, $U_c \in \Re^{n_u \times n_u}$ y $b_c \in \Re^{n_u}$. Luego la codificación x_t teniendo en cuenta la codificación del elemento anterior y h_t quedan definidos por:

$$\begin{aligned} c_t &= f_t c_{t-1} + i_t \tanh(\hat{c}_t) \\ h_t &= o_t \tanh(c_t) \end{aligned} \tag{1.4}$$

1.2.2. Redes Neuronales Convolucionales sobre secuencias

La arquitectura de una CNN (LeCun et al., 1998) sobre una secuencia de texto es capaz de capturar dependencias temporales a corto plazo mediante filtros unidimensional que analizan

n-gramas de palabras o caracteres y cuyos parámetros son compartidos durante cada paso de manera similar a las LSTM.
Esto es, sea $X \in \Re^{l \times d}$ la secuencia de entrada de longitud l donde cada elemento es un vector $d-dimensional$, F_k el conjunto de filtros con ventana k de una capa convolucional, cada uno de los $F_{ki} \in \Re^{k \times d}$ son inicializados de manera independiente y de la misma forma aprenden a capturar relaciones a corto plazo dentro en X. La operación convolución (*conv-op*) transforma a X en una nueva secuencia $X' \in \Re^{(l-k) \times n_f}$ donde $n_f = |F_k|$ de la siguiente forma:

$$X'_{ij} = \sum X_{[i:i+k]} * F_{kj} \quad para \quad j \in [0, F_k - 1],\ i \in [0, l-k] \tag{1.5}$$

Como se puede observar en la Figura 1.2 donde se representa un paso del desplazamiento de la ventana del filtro:

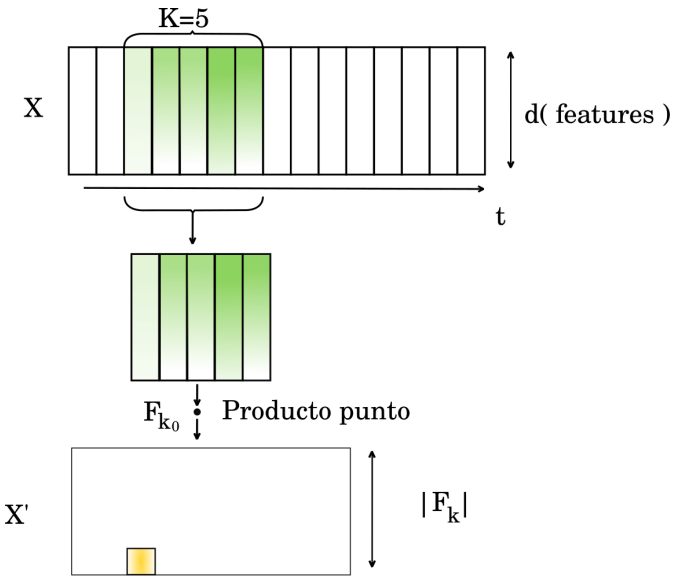


Figura 1.2: Operación convolución sobre X con filtros de tamaño de ventana k.

Vinculada a la *conv-op* en este tipo de arquitectura se suele emplear además una capa de *pooling*, en la cual se combinan los valores de elementos en una vecindad para reducir la dimensionalidad de la secuencia seleccionando los valores más importantes de dicha vecindad, esta operación de *pooling* con una ventana de tamaño k sobre una secuencia X esta definida por:

$$X'_i = f(X_{[i:i+k]}) \quad para \quad i \in [0, l-k] \tag{1.6}$$

Donde f es la función de *pooling*, regularmente empleadas las funciones Max o Avg.

1.2.3. Mecanismos de Atención

Los mecanismos de atención, son técnicas de procesamiento de las entradas de una arquitectura o de algún resultado intermedio que permiten a la red prestar mas "atención" a elementos específicos de una secuencia o establecer la importancia relativa sobre un elemento

del resto. En la práctica, la atención permite a las redes neuronales aproximarse al mecanismo de atención visual que utilizan los humanos.

Self-Attention

Sea x_t la representación del $t-esimo$ elemento de la secuencia, una capa de *self-attention* (auto-atención), captura en una matriz A cuan similar es x_t con sus vecinos. Específicamente $\alpha_{t,t'} \in A$ expresa la relación de x_t con x'_t y de manera similar al de una red recurrente este valor se calcula como:

$$\begin{aligned} g_{t,t'} &= \tanh(W_x x_t + W_{xt'} x_{t'} + b_g) \\ a_{t,t'} &= \sigma(W_a g_{t,t'} + b_a) \end{aligned} \tag{1.7}$$

Donde σ es la función sigmoide, $W_x, W_{xt'} \in \Re^{n_u \times d}$ son las matrices de parámetros encargadas de codificar la información de x y x' para expresar su compatibilidad, $W_a \in \Re^{n_u \times n_u}$ la matriz de parámetros correspondiente a su combinación no lineal y b_g y b_a los correspondientes términos de bias.
A partir de A el estado correspondiente al vector x_t, $\hat{x}_t$ está dado por la suma ponderada de sus elementos vecinos x'_t:

$$\hat{x}_t = \sum_{i=0} a_{t,t'} x_{t'} \tag{1.8}$$

Luego $\hat{x}_t$ expresa cuan atendido debe ser x_t condicionado por el contexto de su vecindad.

Scaled Dot-Product Attention

El mecanismo *Scaled Dot-Product Attention* (Atención con Producto-Punto Escalado) primero se mapea cada elemento de la secuencia con tres representaciones (*query* y un par *key-value*) para calcular el índice de compatibilidad entre cada par de elementos. Luego, para cada x_t es evaluada su compatibilidad con respecto a cada uno de los elementos vecinos relacionando su *query* (q_t) con las *keys* de los vecinos ($k_{t'}$), estos valores de compatibilidad son escalados y normalizados con la función $softmax$ y empleados para ponderar los vectores de *value* (v_t). Finalmente la representación de $\hat{x}_t$ es calculada como la suma ponderada de los v_t. En forma matricial queda expresado como:

$$Attention(Q, V, K) = softmax(\frac{Q \times K^T}{\sqrt{d_k}}) \times V \tag{1.9}$$

Donde $Q, K \in \Re^{n \times d_k}$ y $V \in \Re^{n \times d_v}$ son el resultado del producto de las matrices de parámetros de *query*, *key* y *value* con los elementos de la secuencia respectivamente, por tanto en la fila t contienen el mapeo del elemento x_t y d_k, d_v corresponden a la dimensionalidad de las vectores de *key* y *value* respectivamente.

1.2.4. Transformers

La arquitectura Transformer (Figura 1.3) basada en mecanismos de atención, específicamente *multihead attention* (Vaswani et al., 2017), está diseñada para tratar problemas de

machine translation y la conforman dos módulos, el primero conocido como *Encoder* (Codificador) es alimentado con una secuencia textual y se encarga de encontrar una codificación para cada elemento teniendo en cuenta la información de su contexto. El segundo módulo, *Decoder* produce los elementos de una nueva secuencia en el modelado de lenguaje, haciéndolo de uno a la vez y teniendo en cuenta los elementos generados anteriormente y las codificaciones obtenidas por el Encoder de la secuencia de entrada.
La principal ventaja de las Transformers con respecto a las arquitecturas secuenciales más tradicionales, e.g., GRU y LSTM es que en vez de analizar la información textual en una dirección, esta toma en consideración la entrada completa relacionando cada elemento con el contexto de su vecindad simultáneamente lo cual evita el problema de "memoria" a corto plazo de las RNN. Sin embargo pudiera parecer que existe una pérdida de la percepción del tiempo al analizarlo todo a la vez, es por esto que ademas de representar el texto con un *embedding* de palabras, se tiene en cuenta un *encoding* de posición.

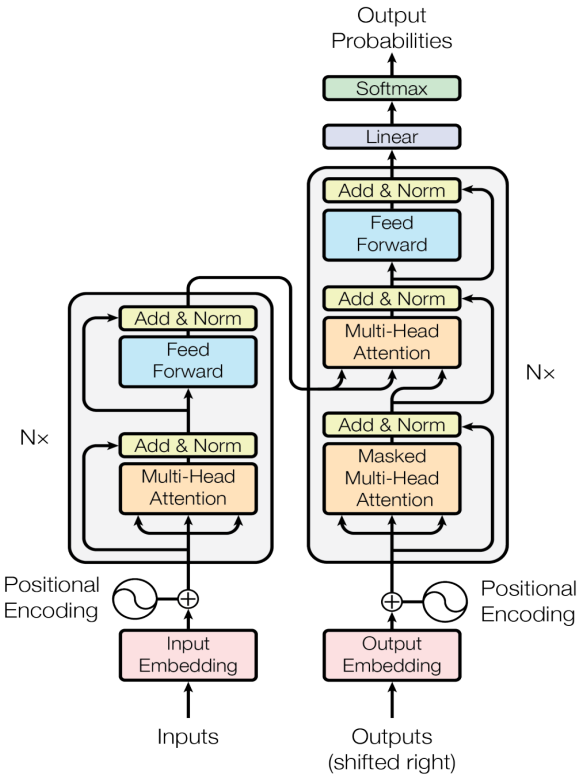


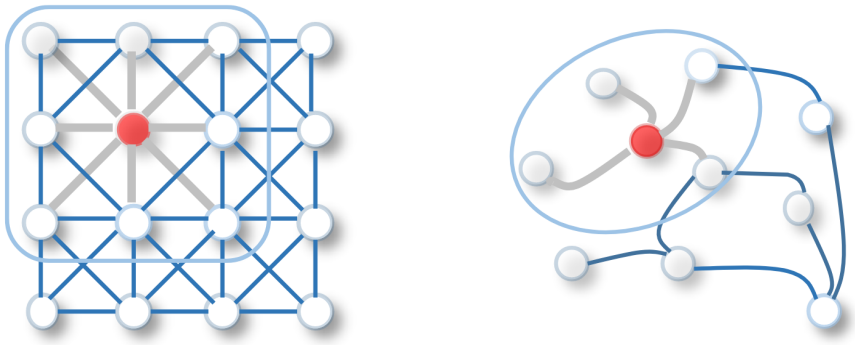
Figura 1.3: Arquitectura Transformer. Módulo Izquierdo, Codificador. Módulo Derecho, Decodificador. (Vaswani et al., 2017)

En la Figura 1.3 se observan cada uno de los módulos los cuales se apilan (N_x), i.e., antes de que el Decoder reciba la informacion codificada de la secuencia de entrada, esta transita por una serie de bloques codificadores haciendo uso de una red residual para prevenir cualquier pérdida de la información.
Dentro de la mayoría de las tareas de NLP este tipo de modelo ha alcanzado nuevos estados del

arte simplemente haciendo *transfer learning* del modulo de Encoder hacia la tarea específica. Este módulo es entrenado en dos tareas: (i) predecir palabras enmascaradas de la secuencia; (ii) dadas dos oraciones, predecir si una está a continuación de la otra en el texto, lo cual hace que el modelo llegue a "entender" el funcionamiento del lenguaje y sea relativamente fácil de entrenar sobre otras tareas como el análisis de sentimientos. Para esta última tarea, dado el par de oraciones como una secuencia, se añade un *token* [CLS] al inicio del cual su estado oculto se toma para realizar la clasificación y otro [SEP] al final de cada oración.

1.2.5. Redes Neuronales Convolucionales en Grafos

La principal diferencia entre las CNNs y las Redes Neuronales Convolucionales en Grafos (*Graph Convolutional Neural Nets* GNN) es que las primeras están diseñadas especialmente para tratar datos regularmente estructurados (Euclideanos), mientras que las GNN son su versión generalizada capaz de extraer información en datos donde no existe una relación de orden entre un nodo y otro y las conexiones entre cada uno de ellos es variable (datos irregulares o datos estructurados no Euclideanos), estas dos propiedades son totalmente opuestas a las relaciones que existen entre los píxeles de una imagen (Figura 1.4).



(a) Datos Estructurados Euclideanos

(b) Datos con Estructura No Euclideana

Figura 1.4: Relación de Estructura en los Datos. (Wu et al., 2021)

Un uso típico de las GNN es la clasificación de nodos en una estructura. En este tipo de problema el nodo v se caracteriza por un conjunto de features x_v y corresponde a una clase t_v. Dado un grafo $G = (V, E)$ parcialmente etiquetado, nuestro objetivo es predecir a que clase corresponde cada nodo no etiquetado. Mediante una GNN, cada nodo comparte información con su vecindad y transforma su representación a un estado h_v de manera que exprese como este pertenece a su contexto. Específicamente,

$$h_v = f(x_v, E_v, H_v, X_v) \tag{1.10}$$

Donde $E_v = \{(i,j)|(i,j) \in E, v \in \{i,j\}\}$, $H_v = \{h_u|u \in \mathcal{N}(v)\}$ con $\mathcal{N}(v)$ el conjunto de nodos de la vecindad de v y $X_v = \{x_u|u \in \mathcal{N}(v)\}$. Dentro del espacio imagen de f es de nuestro interés que cada nodo tenga un estado h_v unívoco, por lo que apoyándose en el teorema del punto fijo (Brown et al., 1988) es posible encontrar en un proceso iterativo de actualizaciones

con determinados parámetros para nuestra función f este h_v contextualizado. El proceso de actualizaciones llevado a cabo dado f es conocido como paso de mensajes. Luego, en forma matricial el estado de todos los nodos en la actualización $t+1$ esta dado por:

$$H^{t+1} = F(H^t, X) \tag{1.11}$$

Con H y X matrices donde a cada fila i le corresponde h_i y x_i respectivamente. Luego, la clasificación de un nodo es llevada a cabo mediante la una nueva función g:

$$t_v = g(h_v, x_v) \tag{1.12}$$

Muchos diseños de la función de agregación han sido estudiados (Kipf and Welling, 2017) teniendo en cuenta propiedades de la topología del grafo y la simetría que debe cumplir el análisis de $\mathcal{N}$ i.e., F no puede ser sensible al orden de agregación de la información. En este trabajo, hacemos uso específicamente de las GNN de tipo espectral (Wu et al., 2021) y la clasificación de G. Este tipo de enfoque no difiere mucho de la clasificación de nodos, pues sigue siendo fundamental la determinación de un estado para cada nodo que exprese su información contextual. Esta información contextual es agregada mediante una operación de *pooling* para alimentar una red densa y proceder con la clasificación de la estructura.

Capítulo 2

Framework

El esquema común para clasificar perfiles teniendo en cuenta cierta característica consiste en: i) extraer rasgos textuales de los documentos del autor, en nuestro caso de los tweets; ii) construir una representación a nivel de tweet o del perfil y finalmente iii) entrenar un modelo de clasificación a partir de la representación elaborada.
Como se puede observar, existe un proceso de identificación de los features empleados para contrastar las características entre tipos de perfiles, haciendo que el desempeño del modelo de clasificación dependa directamente de la robustez de este paso.
Mediante la explotación de modelos de Aprendizaje Profundo, se pretende prescindir de rasgos extraídos manualmente que pudieran resultar ruidosos a los modelos clasificadores o dejar de capturar relaciones claves de la estructura semántica y/o sintáctica, como se expone en la Sección 1.1.

En este Capítulo se describe un *framework* para llevar a cabo el perfilado de autores, basado en el empleo de rasgos abstractos obtenidos a partir de diferentes arquitecturas de DL y la combinación de las mismas. De manera general sus principales contribuciones son:

- Se propone un framework para el perfilado semi-supervisado de autores en redes sociales. El mismo lleva a cabo el aprendizaje de rasgos abstractos para la representación en un espacio latente de los tweets, y construye a partir de los mismos el modelado del perfil.
- Es presentada una combinación de rasgos de estilo a niveles de palabra, oración y otras estructuras gramaticales, para evaluar su influencia sobre la representación exclusiva mediante features extraidos por los modelos de DL.
- Se exponen distintas arquitecturas para modelar tweets y perfiles de usuarios dentro del mismo framework, que relacionen de manera diferente la información para realizar un estudio comparativo. Además se propone una clasificador para combinar estas modelaciones, basado en ML tradicional.

2.1. Rasgos a Nivel de Tweet

El análisis del historial de un perfil de usuario en redes sociales como Twitter, involucra una cantidad considerable de información textual, lo cual puede resultar desafiante para el

entrenamiento de algunos modelos de DL, sobretodo debido a las relaciones a largo plazo que es necesario contemplar a la hora de clasificar un perfil o extraer sus rasgos. Este fenómeno es independiente del nivel de supervisado con que se afronte la tarea y es conocido como *information vanishing* (Hochreiter et al., 2001). Otros modelos son capaces de lidiar con este problema, sin embargo, están limitados por la complejidad temporal que poseen para analizar secuencias de texto, tal es el caso de la popular arquitectura *transformer* donde cada capa de atención tiene $O(n^2 * d)$, con n y d la longitud de la secuencia y la dimensionalidad de sus elementos respectivamente. Por estas razones separar la clasificación del perfil en distintos momentos analizando primero estructuras menos extensas (i.e., los tweets) y luego el perfil completo, constituye una alternativa en el empleo de DL.
El enfoque empleado para este trabajo se apoya en una arquitectura modular para modelar la información textual primeramente a nivel de tweets (*Codificador*) y luego conformar una agregación de los mismos que sirva para describir y clasificar el perfil (*Clasificador*).

2.1.1. CNN - LSTM

El modelo CNN - LSTM (*Convolutional Neural Netowrk - Long Short Term Memory*) representado en la Figura 2.1, recibe como entrada un tweet, analizando la secuencia a nivel de palabra o a nivel de caracteres.

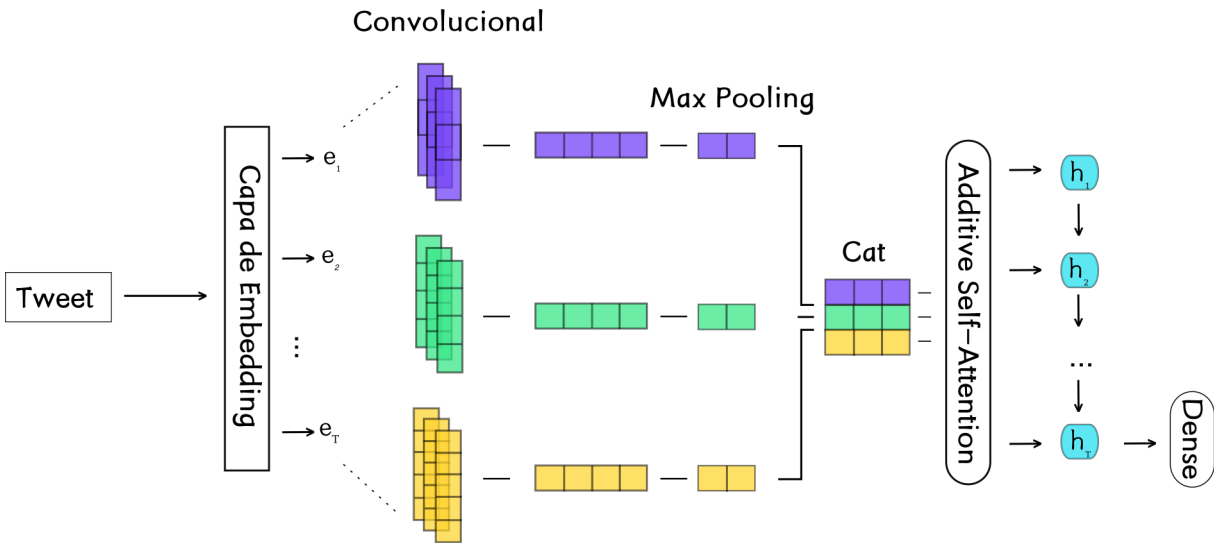


Figura 2.1: Arquitectura de Encoder CNN-LSTM

Cada elemento de la secuencia es codificado por un *embedding* de palabras o caracteres según sea el caso del análisis. En el primero, el modelo emplea un embedding preentrenado con representaciones fijas aprendidas mediante *Word2Vec* de Google (Mikolov et al., 2013). Mientras que a nivel de caracteres estas representaciones son inicializadas de manera aleatoria y aprendidas durante el proceso de entrenamiento del modelo completo.
Como salida de esta capa de embedding obtenemos una matriz de valores reales $E \in \Re^{l \times d}$, con l la longitud de la secuencia y d la dimensionalidad de la representación vectorial de sus

elementos. Luego es aplicada la operación convolución en $1D$ mediante una capa de red convolucional, simulando el análisis de *n-gramas* de palabras o caracteres sobre la representación del embedding que en este punto ya ha capturado ciertas relaciones de contexto entre los elementos del texto.
Esta capa convolucional estará compuesta por filtros de distintos tamaños de ventana (3, 4, 5) inspirada en las Redes de Entradas (*Inception Neural Networks*) (Szegedy et al., 2014) para capturar relaciones espaciales dentro de la secuencia a corto plazo. Para cada tamaño de ventada (i.e., 3, 4 y 5) se emplean además 32 filtros que aprenden estas relaciones de manera independiente.
Una vez efectuada la convolución, sea k el tamaño de la ventana, la longitud de la secuencia queda reducida a $l-k+1$, luego para cada k es aplicada la operación de *max-pooling*, mediante la cual el modelo aprenderá a preservar solamente la información más relevante dentro de un *n-grama*. El pooling se realiza de manera que en el paso $i-esimo$ la ventana no contempla los datos analizados en el paso $(i-1)-esimo$ por lo que el salto es equivalente al tamaño de la ventana, reduciendo nuevamente la secuencia a $\lceil \frac{l-k+1}{k} \rceil$.
La concatenación resultante de la *Inception Network* es considerada como una nueva secuencia de mapeo de rasgos, sin embargo, cada uno sus elementos se relacionan entre si aportando información relevante o no sobre la tarea a la cual responde el entrenamiento del modelo. Esta secuencia es procesada por una capa de *self-attention* (Sección 1.2.3) para ponderar los features teniendo en cuenta su nivel de importancia en vez de hacer a la red neuronal prestar atención de la misma forma a todos los elementos.
Una vez combinada la información a través del mecanismo de atención, la secuencia es explorada y condensada por una LSTM, en la cual el fenómeno de *information vanishing* estará inhibido gracias al mecanismo de atención. De la salida de la LSTM es preservado solamente el último estado oculto en el cual quedan capturadas relaciones a largo plazo en del estilo del autor y contenido del texto, finalmente es transformado a un elemento de un espacio latente por medio de una capa densa. Esta transformación constituye la representación del tweet con el que fue alimentado el modelo.

2.1.2. Codificación basada en Transformers

Teniendo en cuenta que al analizar individualmente los tweets la cantidad de información a procesar es reducida considerablemente y la capacidad de "comprensión" del lenguaje que han demostrado los modelos basados en arquitecturas Transformers (TM), se propone el empleo de modelos pre-entrenados para modelar representaciones abstractas de los tweets. Para ello primeramente se realiza un proceso de refinado (*fine-tuning*) empleando la librería HuggingFace Transformers[1] con datos que respondan a la tarea sobre la cual se llevará a cabo el perfilado.

En el proceso de *fine-tuning* se añade una capa intermedia que recibe los vectores de la secuancia de salida del TM. En esta secuencia de features, se preserva solamente el vector asociado al *token* [CLS]. Luego esta capa intermedia se apila una capa de salida encargada de realizar las predicciones para la tarea sobre la cual se refina el modelo.
Para cada uno de los bloques codificadores del encoder del Transformer es empleado un coeficiente de aprendizaje *learning rate* distinto teniendo en cuenta un refinado gradual-discriminativo de cada bloque, incrementándolo a medida que la red se vuelve mas profunda.

[1] https://huggingface.co/transformers

Esto es, $\alpha_i = \lambda_i \alpha_0$ y $\lambda_i = \lambda_{i-1} + 0.1$, donde α_i representa el *learning rate* del $i-esima$ bloque y λ_i es un multiplicador para determinar α_i a partir de α_0. Empleando este *learning rate* dinámico se preserva la mayor cantidad de informacion aprendida durante el proceso de pre-entrenado el las capas más superficiales, encargadas de capturar rasgos generales, y se sesga el aprendizaje de las capas más profundas hacia la tarea específica abordada en el perfilado.

2.2. Modelado y Clasificación del Perfil

Una vez sintetizada la información abstracta de los tweets, es importante la forma en la que se relaciona cada uno para clasificar el perfil, para ello se propone asumir el conjunto de los tweets como i) una secuencia y ii) una estructura basada en grafos.

2.2.1. Modelado Secuencial. Att-LSTM

Aun cuando no existe necesariamente una relación temporal que exprese particularidades del estilo de escritura del perfil, es posible agregar la información de los tweets mediante una LSTM. Para ello, en el proceso de entrenamiento se construye una secuencia con los tweets, y en cada ciclo de entrenamiento (i.e., *epoch*) de la red neuronal, esta secuencia es permutada aleatoriamente para evitar que el modelo capture erroneamente cualquier relación temporal. La Figura 2.2 muestra la arquitectura general del modelo **Att-LSTM** (*Attention- Long Short Term Memory*) empleada para modelar como secuencia el perfil y clasificarlo.

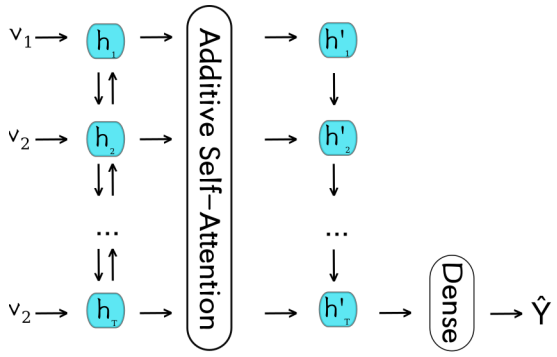


Figura 2.2: Modelado y Clasificación del Perfil. Att-LSTM

El modelo recibe una secuencia donde cada elemento corresponde a los rasgos extraídos para un tweet del perfil, esta secuencia es enviada a una LSTM Bidireccional (Bi-LSTM) (Schuster and Paliwal, 1997), la cual consiste en una célula de LSTM de 64 unidades que calcula para cada elemento dos estados ocultos concatenados, uno de ellos correspondiente a un recorrido de izquierda a derecha y el otro en sentido contrario.
La Bi-LSTM no detecta solamente relaciones de un elemento con los previos en la secuencia sino que también lo relaciona con los que aparecen posteriormente, de esta forma se capturan las relaciones independientes de la posición de cada tweet dentro de la secuencia.
La salida de la Bi-LSTM es enviada a una capa de *self-attention* para emplear la misma estrategia propuesta en la Sección 2.1.1 y luego a una capa de LSTM simple, donde nuevamente se preserva el último estado oculto y se condensa su información en una capa densa. Este

modelado el perfil, finalmente es enviado otra capa densa encargada de clasificar el perfil según la tarea para la que se entrene el modelo.

2.2.2. Modelado basado en Grafo. SGN

Dentro de las redes sociales en muchas ocasiones la información contextual es compartida de manera dispersa, e.g., el contenido de una idea relacionada con la difusión de odio puede ser construido relacionando la información de diferentes posts no necesariamente subidos de manera secuencial. Algunos trabajos han mostrado la importancia del contenido en tareas de AP (Ortega-Mendoza et al., 2018), afirmando que rasgos basados en contenido a veces son más discriminativos que rasgos de estilo (Reddy et al., 2016). Todo esto sugiere la idea de emplear una modelación no euclidiana del conjunto de tweets codificados y compartir la información de un tweets a los otros mientras que su información individual es transformada para expresar como este pertenece a su contexto. Las Redes Neuronales basadas en Grafo están especializadas en aprender patrones de este tipo de representaciones no estructuradas.
El perfil es por tanto modelado con una representación basada en grafos, donde cada nodo es un tweet y cada uno de ellos esta conectado con todos los otros. Luego, este grafo es procesado por una Red Neuronal Convolucional Espectral de Grafos (*Spectral Graph Convolutional Neural Network SGN*).
El hecho de que cada nodo en el grafo modelado este conectado a todos los otros, hace que luego de un ciclo de paso de mensajes, un nodo individual, tenga conocimiento sobre cada nodo en el grafo, empleando el operador convolucional propuesto en (Kipf and Welling, 2017) definido como:

$$X' = ReLU(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}X\Theta) \tag{2.1}$$

Donde X es la matris de representaciones vectoriales de los nodos, $\hat{A} = A + I$ es la matriz de adyacencia A agregada a la matriz identidad I lo que implica que autoconexiones para cada nodo son introducidas a la representación para que en el proceso de agregación se preserve su información original. La matriz D es una matriz diagonal que contiene el grado del nodo $i - esimo$, i.e., $D_{ii} = |V|$ con V el conjunto de los vértices. Finalmente, Θ es la matriz de parámetros de aprendizaje para determinar la codificación de cada nodo con sus rasgos que junto a la Matriz Laplaciana Normalizada expresan el nivel de variación entre los rasgos de los nodos dentro del grafo.
Dada la formulación matricial de esta operación, a nivel de nodo esta queda definida como:

$$x'_i = ReLU\left(\Theta \sum_{j \in \mathcal{N}(i) \cup \{i\}} \frac{1}{\sqrt{\hat{d}_j \hat{d}_i}} x_j\right) \tag{2.2}$$

Aquí, x_i representa la codificación del nodo $i-esimo$, d_i el grado del mismo y $\mathcal{N}(i)$ su conjunto de nodos vecinos.
Como es posible observar en 2.2 y 2.1, en este esquema convolucional la información es compartida simplemente mediante una suma normalizada antes de calcular la nueva codificación x'_i mediante Θ.
En la red convolucional propuesta este proceso de paso de mensaje y actualización es repetido

a través de dos capas convolucionales. Luego, la información de los nodos es combinada a través de una capa de $mean - pool$ para alimentar una capa densa que sintetice la información. La salida de esta última capa es considerada como la modelación del perfil y es enviada a otra capa densa para realizar la clasificación.

2.3. Clasificación basada en ML. Deep Impostor

Un hecho interesante de sobre los modelos de DL es que a pesar de ser potentes a la hora de detectar rasgos abstractos que permitan particionar el espacio de representaciones en diferentes clases, cuando la cantidad de datos disponibles para el entrenamiento no es suficiente, su desempeño se ve afectado. Por esto se propone emplear las modelaciones en un espacio latente de los perfiles descritas en la Sección 2.2.1 y Sección 2.2.2 con el Método Profundo de los Impostores *(Deep Impostor Method DIM)* basado en el Método de los Impostores (Seidman, 2013), para realizar un estudio comparativo del desempeño en la clasificación binaria.

Como en el método original, para realizar una predicción sobre un objeto desconocido (en este caso un perfil de autor) es definido previamente un conjunto H y K de ejemplos positivos y negativos respectivamente y el objeto desconocido u.
De H es muestreado de manera uniforme un subconjunto $\bar{H}$ como prototipos de la clase positiva y es analizado para cada $\bar{H}_i$ si u es mas similar a este prototipo que a los elementos de un conjunto $\bar{K}_i$, i.e., $\mathcal{F}(u, \bar{H}_i) > \mathcal{F}(u, \bar{K}_{ij})$ donde $\mathcal{F}$ es una función de similitud; en este caso coseno. Luego de esto, se dice que u teniendo en cuenta $\bar{H}_i$ es un candidato a pertenecer a una clase positiva según un voto por mayoría, esto es:

$$P_i(u, \bar{H}_i) = \begin{cases} 1 & \text{if } \sum_{j}^{|\bar{K}_i|} [\mathcal{F}(u, \bar{H}_i) > \mathcal{F}(u, \bar{K}_{ij})] > \frac{|\bar{K}_i|}{2} \\ 0 & e.c.o.c \end{cases}$$

Como los rasgos de los objetos serán aprendidos por medio de los modelos de DL descritos, es obviado el procedimiento de selección de rasgos expuesta en el método original, debido a que remover de manera indiscriminada alguno de ellos puede resultar en la destrucción de relaciones de similitud interclase o intraclase.
Luego de definido cada P_i, un perfil u es clasificado según la regla:

$$\hat{y}(u) = \begin{cases} 1 & \text{if } \sum_{i}^{|\bar{H}|} P_i(u, \bar{H}_i) > \frac{|\bar{H}|}{2} \\ 0 & \text{otherwise} \end{cases}$$

2.4. Rasgos Manuales de Estilo

Para la representación del perfil es analizado un conjunto de 177 rasgos que capturen características relevantes del estilo de escritura. Los rasgos son estructurados en seis subconjuntos considerando diferentes capas textuales. Las capas son: booleanas; caracteres; oraciones; párrafo; sintáctica y de documento.

Ejemplos de los elementos de estas capas son:

1. Capa booleana: Usos de la misma palabra para terminar una oración y comenzar la siguiente.
2. Capa de caracter: Media de la longitud de las palabras.
3. Capa de oración: Media de la cantidad de palabras. Media de la cantidad de preposiciones distintas.
4. Capa de párrafo: Media de la cantidad de oraciones. Media de la cantidad de palabras.
5. Capa sintáctica: Proporción de sustantivos y adjetivos.
6. Capa de documento: Media de la longitud de las oraciones.

Esta representación es completamente independiente del genero textual e involucra valores con distintas estructuras de datos como se muestra en la Figura 2.3.

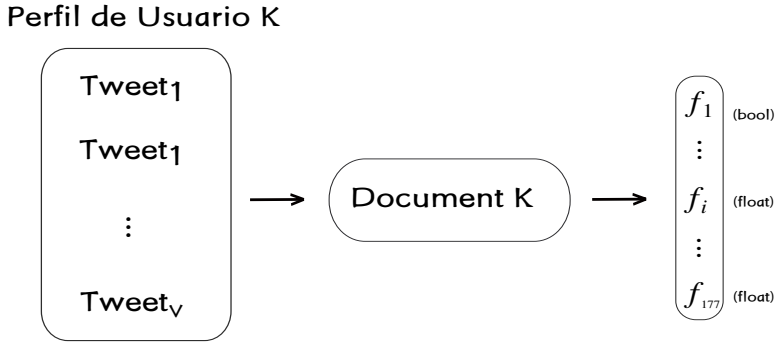


Figura 2.3: Representación de perfil mediante rasgos manuales

Capítulo 3

Marco Experimental

En este capítulo, son descritos los conjuntos de datos sobre los que se evaluará el desempeño de la estrategia de análisis modular propuesta para abordar el perfilado de autores. Luego se exponen las metricas empleadas y finalmente se presentan los resultados experimentales asociados a cada módulo (i.e., Codificador y Clasificador) de manera independiente y a las combinaciones de los mismos.

3.1. Tareas

Las colecciones de datos existente relacionadas con tareas de AP mayormente se encuentran anotadas a nivel de perfil y con información relativa a tareas especificas. Para evaluar los modelos propuestos, siguiendo este esquema, en este trabajo se introducen los conjuntos de datos de las tareas compartidas en las ediciones de 2019-2021 de PAN, cada una de estas propuestas con un enfoque multilingüe, analizando el problema para perfiles de usuarios del idioma ingles y español.

3.1.1. Bots and Gender Profiling at PAN 2019

En la tarea Bots and Gender Profiling, dado conjunto de exactamente 100 posts pertenecientes a un perfil de usuario de Twitter y teniendo en cuenta que no existe ningún tipo anotación a nivel de tweets, debe determinarse si este corresponde a un bot o a un ser humano y para el segundo caso predecirse el genero sexual de la persona.
Para evaluar el desempeño de los modelos propuestos, en nuestro trabajo se prestará atención solamente al problema de discernir entre perfiles de usuarios humano hombres o mujeres, empleando el conjuntos de entrenamiento propuesto y de la prueba del *early bird*. El conjunto de datos de perfiles humanos está construido a partir de cuentas de usuarios tomadas de corpus creados en ediciones previas de la tarea de perfilado de PAN (Rangel et al., 2017, 2018b). Además los datos se encuentran distribuidos uniformemente tanto para el idioma ingles como para el español, entre las clases '*Male*' y '*Female*' como puede ser observado en la Tabla 3.1.

	(EN) Ingles			(ES) Español		
	Female	Male	Total	Female	Male	Total
Training	720	720	1440	520	520	1040
Test	310	310	620	230	230	460
Total	1030	1030	2060	750	750	1500

Tabla 3.1: Distribución de los datos disponibles para la tarea Bots and Gender Profiling at PAN 2019

3.1.2. Profiling Fake News Spreaders on Twitter at PAN 2020

Profiling Fake News Spreaders on Twitter, dentro de PAN 2020, introduce el análisis de rasgos de la personalidad de los autores, proponiendo la tarea de discriminar entre usuarios de Twitter que han compartido noticias falsas de aquellos que nunca lo han hecho, basándose en un conjunto de 100 tweets (carentes de algún tipo de anotación) tomados de su perfil.
El corpus propuesto por los organizadores de la tarea (Rangel et al., 2020b), fue construido seleccionando de sitios web *fact-checking* (comprobadores de hechos) noticias etiquetadas como falsas, luego mediante la búsqueda de tweets relacionados con estas *fake news*, se identificaron a sus correspondientes usuarios como ejemplos positivos de *fake news spreaders* (faker) tomando aquellos con un mayor número de noticias falsas compartidas y teniendo en cuenta que el contenido del tweet no fuera para desmentir la noticia falsa. En el caso de que el usuario no hubiera compartido información relacionada con las noticias falsas identificadas, este fue etiquetado como *real news spreader* (no faker).

El la Tabla 3.2 se muestra la distribución uniforme de los perfiles para los idiomas español e ingles en los que fue compartida la tarea.

	(EN) Ingles			(ES) Español		
	faker	no faker	Total	faker	no faker	Total
Training	150	150	300	150	150	300
Test	100	100	200	100	100	200
Total	250	250	500	250	250	500

Tabla 3.2: Distribución de los datos para la tarea Profiling Fake News Spreaders on Twitter at PAN 2020

3.1.3. Profiling Hate Speech Spreaders on Twitter at PAN 2021

Para esta tarea dado un perfil de usuario, debía ser determinado cuando este corresponde a un autor que ha difundido en el pasado un discurso de odio teniendo en cuenta 200 tweets tomados de su perfil.
El corpus propuesto por los organizadores fue construido considerando usuarios que han empleado palabras con cierto nivel de toxicidad fundamentalmente relacionadas con la misoginia y xenofobia, ademas se inspeccionaron cuentas de usuarios conocidos como *haters* con apariciones en reportes, asi como su red i.e., followers. Luego para estos usuarios identificados,

fueron anotados manualmente los tweets que comunicaban un discurso de odio y finalmente fueron clasificados como *hate speech spreaders* aquellos usuarios con mas de 10 de estos posts. El conjunto de datos esta distribuido uniformemente para las clases divulgador de discurso de odio (*hater*) y no divulgador de discurso de odio (*no hater*) como se muestra en la Tabla 3.3

	(EN) Ingles			(ES) Español		
	hater	no hater	Total	hater	no hater	Total
Training	150	150	300	150	150	300
Test	100	100	200	100	100	200
Total	250	250	500	250	250	500

Tabla 3.3: Distribución de los datos para la tarea Profiling Hate Speech Spreaders on Twitter at PAN 2021

3.2. Resultados Experimentales

La robustez de los sistemas modulares descansan en el optimo aprendizaje de cada uno de los módulos de manera independiente, en nuestro caso del Codificador y el Clasificador. De manera general para este trabajo se introduce una tarea intermedia semisupervisada para entrenar los modelos Codificadores [1], de esta forma los mismos aprenden relaciones del lenguaje dentro de cada tweet sobre la tarea en cuestión.
Para medir el cuan efectivos resulta cada modelo empleamos las métricas *accuracy* (3.1) teniendo en cuenta los datos clasificados clasificados correctamente para los ejemplos positivos y negativos (TP y TN) y los clasificados de manera errónea (FP y FN).

$$acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.1}$$

3.2.1. Codificador CNN - LSTM

Para el análisis secuencial de los tweets de manera independiente, el primer modelo en analizar será CNN - LSTM. Este combina relaciones espaciales detectadas por una 1D-CNN mediante una red recurrente, la cual expresa como relaciones a largo plazo la información de los n-gramas condensados por la CNN como se muestra en la Figura 2.1. Sin embargo para nuestro enfoque es añadido el análisis en paralelo de los niveles sintácticos de palabra y caracteres.

Debido a la finitud de la cardinalidad de los diccionarios en los *embeddings de palabras*, la cual esta dada por la memoria y tiempo disponible para entrenar un modelo de *embedding*,

[1] El objeto de predicción de esta tarea semisupervisada, estuvo condicionado por la carencia de anotación a nivel de tweet en cada uno de los corpus y se relaciona con la pertenencia o no a determinado tipo de perfil.

es recurrente que al analizar textos informales como los tweets se pierda información aportada por elementos que no están presentes en el diccionario, pero que si contienen información semántica y de estilo, ejemplo de ello son las palabras con elongación de caracteres (e.g., holaaaa) y/o *typos*. Por ello empleando la arquitectura descrita en la Sección 2.1.1 introducimos el análisis a nivel de caracteres el cual es combinado con el de palabras a través de una fusión multi-fuente directa[2] enviada a una capa densa de neuronas.

Para entrenar este modelo teniendo en cuenta cada uno de los corpus y tareas de AP en las cuales se medirá su desepeño, se analiza la pertenencia o no de un tweet a clase u otra de perfiles, e.g., si el tweet pertenece a la cuenta de un hombre o una mujer, *hater* o *no hater*, etc. De esta manera, se captura no solo las relaciones inherentes al lenguaje presente en el corups frente a la tarea, si no que se aprende como estas relaciones existen atendiendo a un tipo de perfil u otro.
En el proceso de entrenamiento, el conjunto de datos de entrenamiento de cada uno de los corpus es dividido a una razón de 1:5 en un subconjunto de validación y otro de entrenamiento, de manera que se preserve la distribución uniforme de ejemplos positivos y negativos en cada subconjunto. Para la arquitectura es añadida una neurona en el tope del modelo encargada de responder la probabilidad de que el tweet pertenezca a una clase de cuenta y los pesos del modelo son ajustados por el oprimizador de Estimación Adaptativa del Momento (*Adaptive Moment Estimation*) Adam (Kingma and Ba, 2015) empleando la función de perdida de *Cross-Entropy*.

Durante los experimentos exploramos como afectaba la variación hiper-parametro del coeficiente de aprendizaje $\alpha \in \{1e\text{-}3,\ 15e\text{-}4,\ 2e\text{-}3,\ 3e\text{-}3\}$ en el optimizador de Adam a través de cada tarea, i.e., Bots and Gender Profiling at PAN 2019 (*gender*), Profiling Fake News Spreaders on Twitter (*faker*) y Profiling Hate Speech Spreaders on Twitter (*hater*) como se muestra en la Tabla 3.4 en términos de *accuracy*. Resultando para cada modelo en cada variación de lenguaje valores óptimos de *alpha* entre 1e-3 y 2e-3.

Tarea	(EN) Ingles				(ES) Español			
	1e-3	15e-4	2e-3	3e-3	1e-3	15e-4	2e-3	3e-3
gender	**0.684**	0.68	0.681	0.677	**0.698**	0.693	0.696	0.691
faker	0.734	**0.738**	0.727	0.726	0.741	**0.742**	0.736	0.730
hater	0.696	0.692	**0.698**	0.694	0.611	0.608	**0.611**	0.584

Tabla 3.4: Resultados del entrenamiento para el modelo CNN-LSTM en cada tarea según el coeficiente de aprendizaje α

3.2.2. Codificador Transformer

Para el caso de los codificadores basados en arquitecturas transformers, nuestra propuesta emplea un modelo base para cada idioma, a pesar de que ambos tienen la misma configuracion de BERT-base (Devlin et al., 2018) y son preentrenados siguiendo la misma estrategia de

[2]Simple concatenación de los vectores de cada una de las fuentes, i.e., arquitectura a nivel de palabra y de caracteres

RoBERTa (Liu et al., 2019), i.e., enmascarando de manera aleatoria el 15 % de las palabras de la entrada del modelo en la tarea de modelado de lenguaje enmascarado (Masked Language Modeling MLM).
Para el idioma español se emplea el modelo BETO (Cañete et al., 2020) y para ingles BERTweet (Nguyen et al., 2020), ambos de la biblioteca Transformers de HuggingFace [3]. BERTweet fue preentrenado con un corpus de tweets en ingles, mientras que para BETO , se emplearon textos de otras fuentes como wikis en español, OpenSubtitles y ParaCrawl [4].

En el proceso de finetuning al igual que con la arquitectura CNN - LSTM estos modelos fueron refinados atendiendo a la tarea de predecir cuando un tweet pertenece o no a una clase, siguiendo la estrategia descrita en la Sección 2.1.2 de aplicar un coeficiente α dinámico a medida que se profundiza en la red, el cual ha sido empleado con resultados alentadores por (Palomino and Ochoa-Luna, 2020; Labadie et al., 2021). Los parámetros de los modelos fueron ajustados empleando el optimizador RMSprop (Hinton et al., 2012) con un descenso δ de learning rate lineal por cada epoch, bajo la función de perdida de *Cross-Entropy* y con el mismo esquema de partición de los datos 1:5.
En la Tabla 3.5 se muestra el *accuracy* obtenido para cada tarea en el proceso de refinado, donde se hace evidente un mejor desempeño respecto a los resultados alcanzados con la arquitectura CNN - LSTM de la Tabla 3.4.

Tarea	EN	ES
gender	0.84	0.767
faker	0.65	0.726
hater	0.836	0.805

Tabla 3.5: Finetuning Transformers en cada tarea con $\alpha = 1e\text{-}5,\ \delta = 2e\text{-}5,\ \lambda = 0{,}1$

En particular en ninguna de las dos variantes de idioma sobre la tarea faker, existe tal mejora. Teniendo en cuenta que para la tarea de determinar si un tweet corresponde a un autor *faker* o no, la existencia de sesgo en la representación dado por la información semántica del mensaje podría afectar el rendimiento del modelo, podemos hipotetizar que esta es la causa de la clasificación errónea de algunos tweets.
Por otro lado el desempeño alcanzado en las tareas *gender* y *hater* es razonable dada la forma en que el mecanismo de atención empleado relaciona la información y a la cantidad de parámetros de estos modelos, lo cual permite que cada neurona preste atención a estructuras particulares del lenguaje. Esto último pudiera resultar en un sobreajuste a los datos de entrenamiento de no ser por la enorme cantidad datos de diversas fuentes con las que fueron preentrenados y el α suave empleado en el proceso de *finetuning*.

3.2.3. Clasificador Att-LSTM

Al igual que en los módulos de codificación, este modelo fue entrenado de manera independiente para cada idioma, mediante el optimizador de Adam, de manera que exploramos para cada representación como su combinación con los rasgos de estilo (Sección 2.4) podría

[3] https://huggingface.co/transformers
[4] https://github.com/josecannete/spanish-corpora

afectar el desempeño del modelo dado un coeficiente de aprendizaje α. Esta fusión con los rasgos de estilo fue llevada a cabo introduciendo una capa densa en el modelo para condensar la información del vector de estilo, luego su salida fue concatenada con la salida de la LSTM que contiene la información aprendida directamente de la representación en el espacio latente de los tweets que componen el perfil, siguiendo el flujo del modelo explicado en la Sección 2.2.1.

En la Tabla 3.6 se muestran los resultados de dicho entrenamiento, evaluado sobre el conjunto de datos de prueba propuesto por los organizadores de cada tarea, donde se puede observar que las modelaciones de los perfiles tomando como elementos de la secuencia tanto de un codificador como del otro arrojaron resultados alentadores. Además en las tareas relacionadas directamente con dispositivos comunicacionales, fundamentalmente en la tarea *hater*, los rasgos de estilo tuvieron un impacto positivo en el *accuracy* del modelos, sin embargo, para la tarea *gender* estos rasgos de estilo al parecer resultaron ruidosos o no tuvieron influencia en el modelado del perfil por parte de Att-LSTM al emplear ninguna de las dos representaciones aprendidas por los modelos de DL.

Tarea	α	(EN) Ingles				(ES) Español			
		C1	C2	C2-STY	C1-STY	C1	C2	C2-STY	C1-STY
gender	5e-4	0.797	0.758	0.761	0.800	0.630	0.648	0.650	0.630
	1e-3	0.795	**0.803**	0.789	0.797	0.632	**0.693**	0.660	0.639
	2e-3	0.797	0.802	0.782	0.793	0.635	0.683	0.621	0.637
	3e-3	0.801	0.753	0.779	0.802	0.632	0.667	0.650	0.632
faker	5e-4	0.690	0.645	0.635	0.694	0.785	0.805	0.800	0.780
	1e-3	0.680	0.650	0.645	0.700	0.784	0.805	0.805	0.784
	2e-3	0.690	0.660	0.665	**0.704**	0.775	0.810	0.815	0.795
	3e-3	0.685	0.645	0.655	0.695	0.770	0.810	**0.820**	0.779
hater	5e-4	0.650	0.700	**0.740**	0.660	0.800	0.789	0.798	0.790
	1e-3	0.650	0.730	0.700	0.660	0.790	0.800	0.798	**0.810**
	2e-3	0.700	0.709	0.710	0.670	0.780	0.800	**0.810**	**0.810**
	3e-3	0.649	0.740	**0.750**	0.670	0.790	0.790	0.790	**0.810**

Tabla 3.6: Entrenamiento del modelo Att-LSTM basado en el análisis secuencial del perfil para la combinación de rasgos de los modelos: (i) codificador CNN - LSTM (C1), (ii) codificador transformers y rasgos de estilo (STY)

3.2.4. Clasificador SGN

Para el entrenamiento de la Spectral Graph Network, seguimos un esquema similar al del modelo Att-LSTM, introduciendo una capa densa que condense la información de los rasgos de estilo para concatenar su salida con el pooling descrito en la Sección 2.2.2 con la información de los nodos que componen el grafo y luego siguiendo el flujo de la red. Para optimizar los parametros de aprendizaje nuevamente empleamos Adam evaluando coeficientes de aprendizaje en el espacio [1e-4, 2e-4, 5e-4, 1e-3, 2e-3]. En la Tabla 3.7, se muestran los resultados de las combinaciones más acertadas sobre este modelo.

Tarea	α	(EN) Ingles				(ES) Español			
		C1	C2	C2-STY	C1-STY	C1	C2	C2-STY	C1-STY
gender	1e-4	0.794	0.718	0.750	0.798	0.635	0.610	0.617	0.637
	2e-4	0.797	0.758	0.769	0.798	0.635	0.624	0.622	0.637
	5e-4	0.805	0.760	0.784	**0.806**	0.632	**0.643**	0.624	0.639
	1e-3	0.805	0.765	0.784	0.803	0.630	0.626	0.620	0.635
faker	1e-4	0.690	0.665	0.660	0.695	0.750	0.805	0.810	0.755
	2e-4	0.695	0.650	0.660	**0.710**	0.750	0.810	0.815	0.765
	5e-4	0.690	0.650	0.650	0.705	0.770	0.815	0.815	0.780
	1e-3	0.689	0.654	0.670	0.710	0.765	0.815	**0.820**	0.780
hater	1e-4	0.670	**0.740**	0.730	0.670	0.800	0.780	0.790	0.810
	2e-4	0.670	**0.740**	0.720	0.670	0.800	0.780	0.790	0.800
	5e-4	0.650	0.730	0.709	0.660	0.800	0.800	0.790	0.810
	1e-3	0.640	0.720	0.710	0.670	0.800	**0.820**	0.810	0.810

Tabla 3.7: Entrenamiento del modelo SGN basado en el análisis secuencial del perfil para la combinación de rasgos de los modelos: (i) codificador CNN - LSTM (C1), (ii) codificador transformers y rasgos de estilo (STY)

Como es apreciable, a diferencia del modelo Att-LSTM en la tarea *hater* el uso de los rasgos de estilo no aportaron una mejora al desempeño del modelo, lo cual puede ser causado por la forma en la que la información de todo el grafo es agregada mediante el operador convolucional. Sin embargo, al igual que en el resto de las tareas, la diferencia con respecto al uso de estos rasgos ofrece una evidencia muy discreta sobre lo ruidoso que pudieran resultar estos rasgos sobre el modelo.

3.2.5. Clasificador Deep Impostor

Para este modelo basados en la cardinalidad del conjunto de datos de entrenamientos variamos la proporción representada por los subconjuntos prototípicos $\bar{H}$ y $\bar{K}$ para las clases positivas y negativas respectivamente en el rango de 0.1 a 0.5 con un incremento de 5 unidades porcentuales. Esto ultimo, teniendo en cuenta la complejidad temporal que implica por cada elemento del conjunto de datos de prueba muestrear aleatoriamente la cantidad prefijada de prototipos negativos por cada prototipo positivo, ademas de la complejidad temporal introducida por el calculo de la similitud coseno sobre dos vectores 64-dimensional como el de los perfiles de usuarios en el espacio latente.
El criterio para seleccionar la combinación de rasgos empleados por los modelos (i.e., CGN y Att-LSTM) para modelar los perfiles estuvo condicionado por el desempeño de los mismos a la hora de predecir cuando los perfiles pertenecen a la clase positiva o negativa en cada tarea según se describe en las Secciones 3.2.4 y 3.2.3. Tomando como modelado del perfil la salida de la capa intermedia, previa a la neurona encargada de realizar la calificaron. La Tabla 3.8 muestra para cada tarea los mejores resultados en accuracy, empleando una proporción p.

	(EN) Ingles		(ES) Español	
	Att-LSTM	SGN	Att-LSTM	SGN
gender	0.524(0.30)	**0.780**(0.25)	0.515(0.30)	**0.633** (0.25)
faker	0.495(0.35)	**0.710**(0.25)	0.560(0.35)	**0.805**(0.25)
hater	0.720(0.40)	**0.750**(0.30)	0.790(0.40)	**0.800**(0.30)

Tabla 3.8: Resultados del método Deep Imposotor. Los valores dentro de los paréntesis indican la proporción representada por $\bar{H}$ y $\bar{K}$.

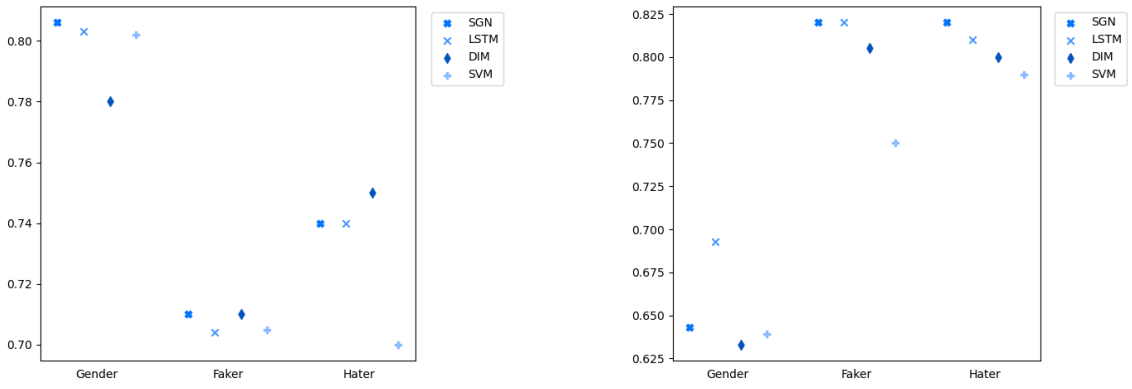
Como se puede observar, los mejores resultados fueron obtenidos con las representaciones determinadas a partir del modelado del perfil como un grafo con la red convolucional espectral, sin embargo en todos los casos exceptuando el idioma ingles en la tarea hater, estuvo por debajo del accuracy de la predicción del modelo SGN para el perfil, lo cual demuestra la robustez de los modelos empleados ante este método.

Finamente, para comparar el desempeño de los modelos propuestos de Deep Learning, se construyo para cada perfil una representación vectorial mediante *tf-idf* sobre BoW, con la cual se entreno una SVM, obteniéndose los resultados mostrados en la Tabla 3.9.

Tarea	EN	ES
gender	0.802	0.639
faker	0.705	0.750
hater	0.700	0.790

Tabla 3.9: Resultados de SVM sobre BoW y tf-idf

Nuevamente para este modelo de Machine Learning los resultados estan por debajo de los alcanzados con la arquitectura modular propuesta, como se puede observar en la Figura 3.1.



(a) Idioma Ingles

(b) Idioma Español

Figura 3.1: Resultados conjuntos de los modelos propuestos para el perfilado de autores en términos de accuracy

Con respecto a las competiciones de PAN 2020 y 2021, sobre la cual tenemos las colecciones

completas y las tareas no fueron evaluadas de manera parcial, podemos decir que nuestros modelos alcanzan resultados competitivos y para el caso de PAN 2020 supera el desempeño medio del mejor modelo de Deep Learning (Giglou et al., 2020) posicionado con 74.25 frente a 76.5 alcanzado por el modelado basado en grafos del perfil propuesto en este trabajo bajo una arquitectura modular, además, para el idioma español alcanzó el mejor accuracy registrado en la competencia.

Para la tarea Profiling Hate Speech Spreaders on Twitter en PAN 2021 nuestra arquitectura SGN alcanzó un certeza media de 78 % frente al 79 % alcanzado frente al mejor modelo, sin embargo, cabe mencionar que nuestra modificación basada en el Método del los Impostores, alcanzó un desempeño, superior para el idioma ingles con respecto a (Siino et al., 2021) (i.e.,73+1).

Estos resultados, dan evidencia de la robustez de las arquitecturas modulares propuestas y en especial de la representación natural de los perfiles como datos no estructurados a través de los grafos.

Conclusiones

En este trabajo de tesis se abordo la tarea de perfilado de autores con un enfoque multilingüe, analizando el caso de los idiomas español e ingles, mediante la propuesta de una arquitectura modular que divide el problema de clasificar un perfil de usuario de Twitter en:
(i) Determinar una representación para cada tweet de manera individual en un espacio latente, empleando una de dos técnicas; la combinación del análisis a nivel de palabra y caracteres a través de redes neuronales convolucionales y de memoria a largo-corto plazo, o modelos basados en la arquitectura transformers.
(ii) Modelar el perfil de usuario para clasificarlo empleando una de dos interpretaciones; la primera consiste en tomar el conjunto de tweets dentro del perfil como una secuencia y la otra consiste en interpretar el perfil como un grafo completo donde todos los nodos se relacionan entre si y mediante una red neuronal convolucional espectral, aprender como la representación de un tweet pertenece a su contexto para clasificar el grafo.
Además se propuso el uso de un conjunto de rasgos de estilos para estudiar como estos influyen en la precisión de los modelos de Deep Learing propuestos para clasificar los perfiles de usuario y se introdujo una variación al método de los impostores empleado anteriormente en tareas de verificación de autoría. Con esta modificación el método fue capaz de manejar representaciones determinadas a partir de model basados en DL y se convirtió la tarea de clasificación de un perfil a una tarea de verificación, donde a través de un conjunto de prototipos se asigno una clase u otra a dicho perfil.
Los resultados experimentales conducidos mostraron la robustez de los modelos independientemente del lenguaje, frente a las propuestas de Deep Learning del estado del arte, en especial la modelación a basada en grafo del perfil empleando las representaciones latentes de las arquitecturas transformers obtuvo resultados alentadores. Por otra parte se identifico la influencia positiva de los rasgos de estilo propuestos para tareas que no involucraban dispositivos del lenguaje complejos como el odio y la ofensa, si no que dependían mas en la forma en la que se comunicaba la información, tal es el caso de la tarea de determinar el genero sexual o la tendencia a postear noticias falsas por parte de los usuarios.

Teniendo en cuenta todo esto, podemos afirmar que nuestro trabajo ha cumplido con sus objetivos y pretendemos introducir los siguientes elementos para mejorar el accuracy de nuestra arquitectura modular:

- Puesto que el modelado basado grafo parece ser una técnica prometedora para expresar aquellas relaciones no estructuradas como las existentes entre los posts, se pretende analizar y construir una representación que capture relaciones mas fuertes considerando las predicciones que produzcan los modelos Codificadores sobre los tweets y la similitud entre los mismos.

- Como alternativa a una construcción mas costosa en espacio de memoria y tiempo de una estructura de grafos mas complejas, se puede seguir considerando el grafo completo con una función de agregación para el paso de mensajes, basada en modelos de atención que permita al SGN discernir cuales tweets de una vecindad son mas importantes a la hora de determinar la representación contextual.
- Para nuestra modificación del Método de los Impostores, explorar métodos para la selección de prototipos que permitan hacer de esta arquitectura una modelo end-to-end, así como construir un sub-modelo de aprendizaje de métrica que sustituya a la función coseno empleada en nuestra propuesta.

Referencias Bibliográficas

Nipun Agarwala, Yuki Inoue, and Alex Sly. 2017. Music composition using recurrent neural networks. *CS 224n: Natural Language Processing with Deep Learning, Spring*.

Md Zahangir Alom, Tarek M. Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C. Van Essen, Abdul A. S. Awwal, and Vijayan K. Asari. 2019. A State-of-the-Art Survey on Deep Learning Theory and Architectures. *Electronics*, 8(3).

Oleg Bakhteev, Aleksandr Ogaltsov, and Petr Ostroukhov. 2020. Fake News Spreader Detection Using Neural Tweet Aggregation—Notebook for PAN at CLEF 2020. In *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Arup Baruah, Kaushik Amar Das, Ferdous Ahmed Barbhuiya, and Kuntal Dey. 2020. Automatic Detection of Fake News Spreaders Using BERT—Notebook for PAN at CLEF 2020. In *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Angelo Basile, Gareth Dwyer, Maria Medvedeva, Josine Rawee, Hessel Haagsma, and Malvina Nissim. 2017. N-GrAM: New Groningen Author-profiling Model—Notebook for PAN at CLEF 2017. In *CLEF 2017 Evaluation Labs and Workshop – Working Notes Papers, 11-14 September, Dublin, Ireland*. CEUR-WS.org.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

R.F. Brown, Calif.) Conference on Fixed Point Theory (1986, Berkeley, American Mathematical Society, and Calif.) International Congress of Mathematicians (1986, Berkeley. 1988. *Fixed Point Theory and Its Applications: Proceedings of a Conference Held at the International Congress of Mathematicians, August 4-6, 1986*. Contemporary mathematics - American Mathematical Society. American Mathematical Society.

Jakab Buda and Flora Bolonyai. 2020. An Ensemble Model Using N-grams and Statistical Featuresto Identify Fake News Spreaders on Twitter—Notebook for PAN at CLEF 2020. In *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish Pre-Trained BERT Model and Evaluation Data. In *PML4DC at ICLR 2020*.

Andrea Cimino, Felice Dell'Orletta, and Malvina Nissim. 2020. TAG-it@ EVALITA 2020: Overview of the Topic, Age, and Gender Prediction Task for Italian. In *Proceedings of the 7th*

evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020), Online. CEUR. org. CEUR Workshop Proceedings (CEUR-WS.org). Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, EVALITA 2020 ; Conference date: 17-12-2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, abs/1810.04805.

Rafael Felipe Sandroni Dias and Ivandré Paraboni. 2019. Combined CNN+RNN bot and gender profiling. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Hamed Babaei Giglou, Jafar Razmara, Mostafa Rahgouy, and Mahsa Sanaei. 2020. LSACoNet: A Combination of Lexical and Conceptual Features for Analysis of Fake News Spreaders on Twitter—Notebook for PAN at CLEF 2020. In *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. 2012. Lecture 6a overview of mini–batch gradient descent. *Coursera Lecture slides https://class. coursera. org/neuralnets-2012-001/lecture,[Online.*

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Aarish Iyer and Soroush Vosoughi. 2020. Style Change Detection Using BERT—Notebook for PAN at CLEF 2020. In *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Fredrik Johansson. 2019. Supervised Classification of Twitter Accounts Based on Textual Content of Tweets. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Youngjun Joo and Inchon Hwang. 2019. Author Profiling on Social Media: An Ensemble Learning Approach using Various Features. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Roberto Labadie, Mariano Jason Rodirguez Cisneros, Reynier Ortega Bueno, and Paolo Rosso. 2021. A Transformer-based Approach for Detecting and Rating Humor and Offense. In *Proceedings of the 15th International Workshop on Semantic Evaluation.*

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Jesus López-Santillán, Luis Carlos González-Gurrola, Manuel Montes-y-Gómez, Graciela Ramírez Alonso, and Olanda Prieto-Ordaz. 2019. An Evolutionary Approach to Build User Representations for Profiling of Bots and Humans in Twitter. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Matej Martinc, Iza Škrjanec, Katja Zupan, and Senja Pollak. 2017. PAN 2017: Author Profiling - Gender and Language Variety Prediction—Notebook for PAN at CLEF 2017. In *CLEF 2017 Evaluation Labs and Workshop – Working Notes Papers, 11-14 September, Dublin, Ireland*. CEUR-WS.org.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

M. Newman, Carla J. Groom, Lori D. Handelman, and J. Pennebaker. 2008. Gender differences in language use: An analysis of 14,000 text samples. *Discourse Processes*, 45:211 – 236.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

R. M. Ortega-Mendoza, A. P. López-Monroy, A. Franco-Arcega, and M. Montes y Gómez. 2018. Emphasizing personal information for author profiling: New approaches for term selection and weighting. *Knowl. Based Syst.*, 145:169–181.

Daniel Palomino and José Ochoa-Luna. 2020. Palomino-Ochoa at SemEval-2020 Task 9: Robust System Based on Transformer for Code-Mixed Sentiment Classification. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 963–967, Barcelona (online). International Committee for Computational Linguistics.

James W. Pennebaker, Ryan Boyd, Kayla Jordan, and Kate Blackburn. 2015. *The development and psychometric properties of LIWC2015*. University of Texas at Austin.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Juraj Petrík and Daniela Chudá. 2019. Bots and gender profiling with convolutional hierarchical recurrent neural network. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Juan Pizarro. 2019. Using N-grams to detect Bots on Twitter. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Juan Pizarro. 2020. Using N-grams to detect Fake News Spreaders on Twitter—Notebook for PAN at CLEF 2020. In *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Francisco Rangel, Anastasia Giachanou, Bilal Ghanem, and Paolo Rosso. 2020a. Overview of the 8th Author Profiling Task at PAN 2020: Profiling Fake News Spreaders on Twitter. In *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Francisco Rangel, Manuel Montes-y-Gómez, Martin Potthast, and Benno Stein. 2018a. Overview of the 6th Author Profiling Task at PAN 2018: Cross-domain Authorship Attribution and Style Change Detection. In *CLEF 2018 Evaluation Labs and Workshop – Working Notes Papers, 10-14 September, Avignon, France*. CEUR-WS.org.

Francisco Rangel and Paolo Rosso. 2019. Overview of the 7th Author Profiling Task at PAN 2019: Bots and Gender Profiling. In *CLEF 2019 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Francisco Rangel, Paolo Rosso, Bilal Ghanem, and Anastasia Giachanou. 2020b. Profiling fake news spreaders on twitter.

Francisco Rangel, Paolo Rosso, Manuel Montes-y Gómez, Martin Potthast, and Benno Stein. 2018b. Overview of the 6th author profiling task at pan 2018: multimodal gender identification in twitter. *Working Notes Papers of the CLEF*, pages 1–38.

Francisco Rangel, Paolo Rosso, Martin Potthast, and Benno Stein. 2017. Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. *Working notes papers of the CLEF*, pages 1613–0073.

Francisco Rangel, GLDLP Sarracén, BERTa Chulvi, Elisabetta Fersini, and Paolo Rosso. 2021. Profiling hate speech spreaders on twitter task at pan 2021. In *CLEF*.

T Raghunadha Reddy, B Vishnu Vardhan, and P Vijaypal Reddy. 2016. A survey on authorship profiling techniques. *International Journal of Applied Engineering Research*, 11(5):3092–3102.

Youssra Riahi and S. Riahi. 2018. Big Data and Big Data Analytics: Concepts, Types and Technologies. *International Journal of Research and Engineering*, 5:524–528.

Paolo Rosso, Martin Potthast, Benno Stein, Efstathios Stamatatos, Francisco Rangel, and Walter Daelemans. 2019. Evolution of the pan lab on digital text forensics. In Nicola Ferro and Carol Peters, editors, *Information Retrieval Evaluation in a Changing World: Lessons Learned from 20 Years of CLEF*, pages 461–485. Springer International Publishing, Cham.

Paolo Rosso and Francisco Rangel Pardo. 2020. Author Profiling Tracks at FIRE. *FIRE 10th Anniversary, SN Computer Science*, 1.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. 2006. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 45(11):2673–2681.

H. A. Schwartz, J. Eichstaedt, Margaret L. Kern, Lukasz Dziurzynski, Stephanie Ramones, Megha Agrawal, Achal Shah, M. Kosinski, D. Stillwell, M. Seligman, and L. Ungar. 2013. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PLoS ONE*, 8.

Shachar Seidman. 2013. Authorship verification using the impostors method. In *CLEF 2013 Evaluation labs and workshop–Working notes papers*, pages 23–26. Citeseer.

Marco Siino, Elisa Di Nuovo, Ilenia Tinnirello, and Marco La Cascia. 2021. Detection of Hate Speech Spreaders using Convolutional Neural Networks. In *CLEF 2021 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. Going deeper with convolutions.

Alex I. Valencia-Valencia, Helena Gómez-Adorno, Christopher Stephens Rhodes, and Gibran Fuentes Pineda. 2019. Bots and Gender Identification Based on Stylometry of Tweet Minimal Structure and n-grams Model. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24.