BOISE STATE UNIVERSITY

Distributed blinding for distributed ElGamal re-encryption

Alejandro Anzola Ávila

October 16, 2019

Boise State University

Definitions

Motivation

This paper shows a protocol for interacting *distributed services* that emphasizes on **step flexibility** rather than evaluate on quantitative measures, such as number of messages exchanged or total computing time.

ElGamal public key encryption

Is based on large prime numbers p and q such that

$$p = 2q + 1$$

Let $\mathcal{G}_p$ be a cyclic subgroup (of order q) of $\mathbb{Z}_p^* = \{i \mid 1 \leq i \leq p-1\}$, where g is a generator of $\mathcal{G}_p$.
Any $k \in \mathbb{Z}_q^*$ can be an ElGamal private key and $K = (p, q, g, y)$ is the public key, and $y = g^k \mod p$.

ElGamal public key encryption

An ElGamal ciphertext $E(m)$ for plaintext $m \in \mathcal{G}_p$ is a pair (g^r, my^r) with r uniformly and randomly chosen from $\mathbb{Z}_q^*$.
Ciphertext $E(m) = (a, b)$ is decrypted by computing b/a^k.

$$b/a^k = my^r/(g^r)^k = m(g^k)^r/(g^r)^k = m$$

When $E(m, r)$ is shown, the value of r is made explicit. Therefore $\mathcal{E}(m)$ is the set $\{E(m, r) \mid r \in \mathbb{Z}_q^*\}$ of all possible ciphertexts for m.

ElGamal public key encryption properties

Given $E(m_1) = (a_1, b_1)$, $E(m_2) = (a_2, b_2)$ and $E(m) = (a, b)$, we have

- $E(m)^{-1} = (a^{-1}, b^{-1})$
- $m' \cdot E(m) = (a, m', b)$
- $E(m_1) \cdot E(m_2) = (a_1 a_2, b_1 b_2)$

The following properties hold

ElGamal Inverse	$E(m)^{-1} \in \mathcal{E}(m^{-1})$
ElGamal Juxtaposition	$m' \cdot E(m, r) = E(m'm)$
ElGamal Multiplication[1]	If $r_1 + r_2 \in \mathbb{Z}_q^*$ then $E(m_1, r_1) \times E(m_2, r_2) \in \mathcal{E}(m_1 m_2)$

[1] Homomorphic property

Re-encryption and Distributed Blinding protocols

Re-encryption protocol

The basic re-encryption protocol is

1. Pick a random[2] $\rho \in \mathcal{G}_p$, then compute $E_A(\rho)$ and $E_B(\rho)$
2. Compute blinded ciphertext $E_A(m\rho) \coloneqq E_A(m) \times E_A(\rho)$
3. Employ threshold decryption to obtain blinded plaintext $m\rho$ from blinded ciphertext $E_A(m\rho)$.
4. Compute $E_B(m) \coloneqq m\rho \cdot E_B(\rho)^{-1}$

[2] The possibility of compromised servers makes computing ρ, $E_A(\rho)$, and $E_B(\rho)$ trickier.

Distributed blinding protocol

Given two related public keys K_A and K_B, the distributed blinding protocol must satisfy the following correctness requirements:

- **Randomness-Confidentiality**: Blinding factor $\rho \in \mathcal{G}_p$ is chosen randomly and kept confidential from the adversary.
- **Consistency**: The protocol outputs a pair of ciphertexts $E_A(\rho)$ and $E_B(\rho)$.

Notation

Symbol	Description
S	Service
n	Number of services
K_S	Service S public key
k_S	Service S private key
(n, f)	Threshold cryptography scheme
m	Plaintext message
ρ	Blinding factor

Marco teórico

Teorema de Bayes

Para variables aleatorias x e y, se tiene que la probabilidad condicional $P(\text{y} \mid \text{x})$ es definida como

$$P(\text{y} \mid \text{x}) = \frac{P(\text{x} \mid \text{y})P(\text{y})}{P(\text{x})}$$

Clasificador Naïve Bayes

Un clasificador de Naïve Bayes estima la probabilidad condicional de las clases por medio de suponer que los atributos son condicionalmente independientes, dado la etiqueta de clasificación y. Donde cada conjunto de d atributos $\mathbb{X} = \{x_1, \ldots, x_d\}$ se tiene

$$P(\mathbb{X} \mid \mathrm{y} = y) = \prod_{i=1}^{d} P(x_i \mid \mathrm{y} = y)$$

El clasificador computa la probabilidad posterior para cada clase y como

$$P(\mathrm{y} \mid \mathbb{X}) = \frac{P(\mathrm{y}) \prod_{i=1}^{d} P(x_i \mid \mathrm{y})}{P(\mathbb{X})} \Rightarrow P(\mathrm{y}) \prod_{i=1}^{d} P(x_i \mid \mathrm{y})$$

Nota Puede ignorarse $P(\mathbb{X})$ debido a que es un termino constante. Para esto se realiza una normalización con una constante ϵ de forma que $\sum_{\forall \mathrm{y} \in \mathbb{Y}} \epsilon^{-1} P(\mathrm{y} \mid \mathbb{X}) = 1$.

Clasificación con Support Vector Machines (SVM)

Técnica de **clasificación** con una frontera de decisión en forma de hiper-planos que permiten aplicaciones con vectores de alta dimensionalidad.

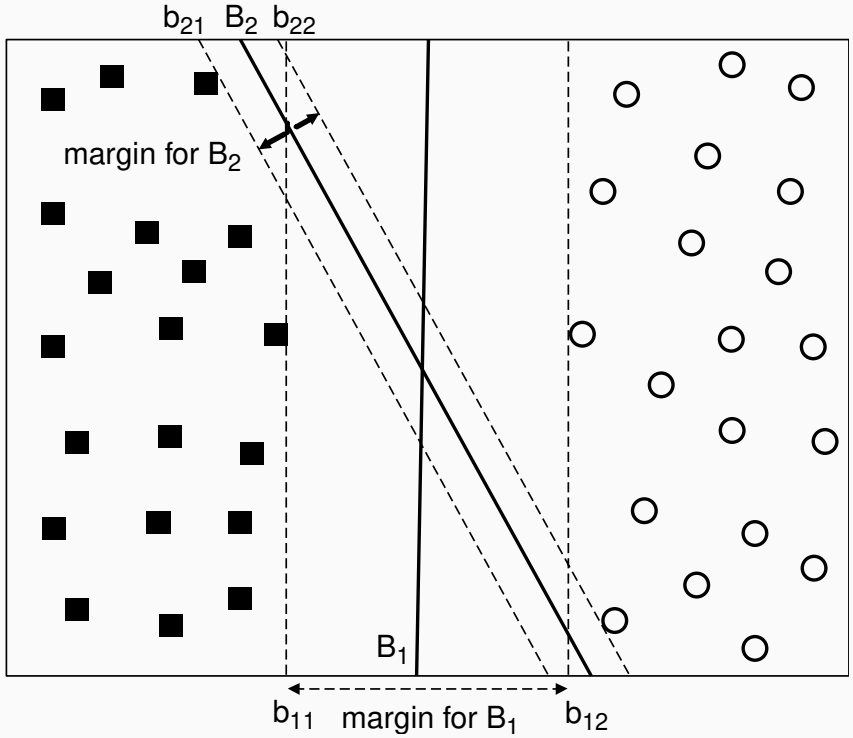

Figure 1: Maximum Margin Hyperplanes. Tomado de [5].

Self-organizing Maps (SOM)

Es un mapa discreto de o neuronas con vectores $\boldsymbol{w} \in \mathbb{R}^m$ que se adaptan a una entrada de $\boldsymbol{X} \in \mathbb{R}^{m \times N}$ de N patrones. Tiene una adaptación con una tasa de aprendizaje α_t y un área de afectación σ_t que se reducen por cada iteración $t \in \{0, \ldots, T\}$.

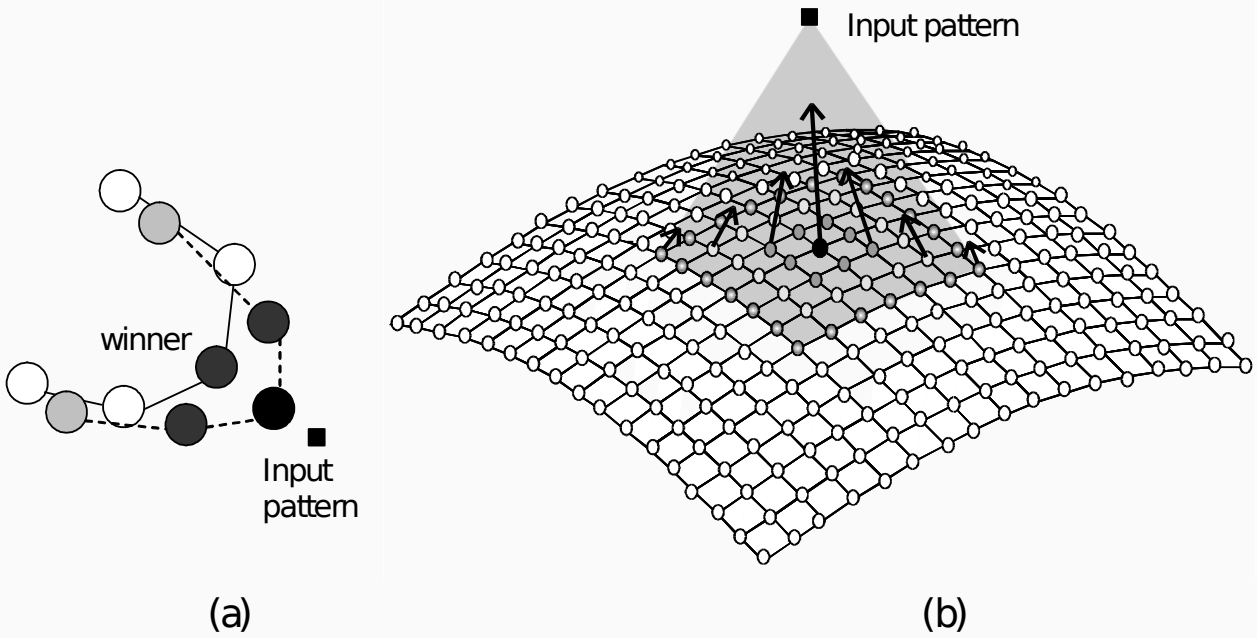

Figure 2: Proceso de adaptación de SOM, (a) uni–dimensional, (b) bi–dimensional. Tomado de [2].

Aplicación de SOM en perfilamiento de criminales

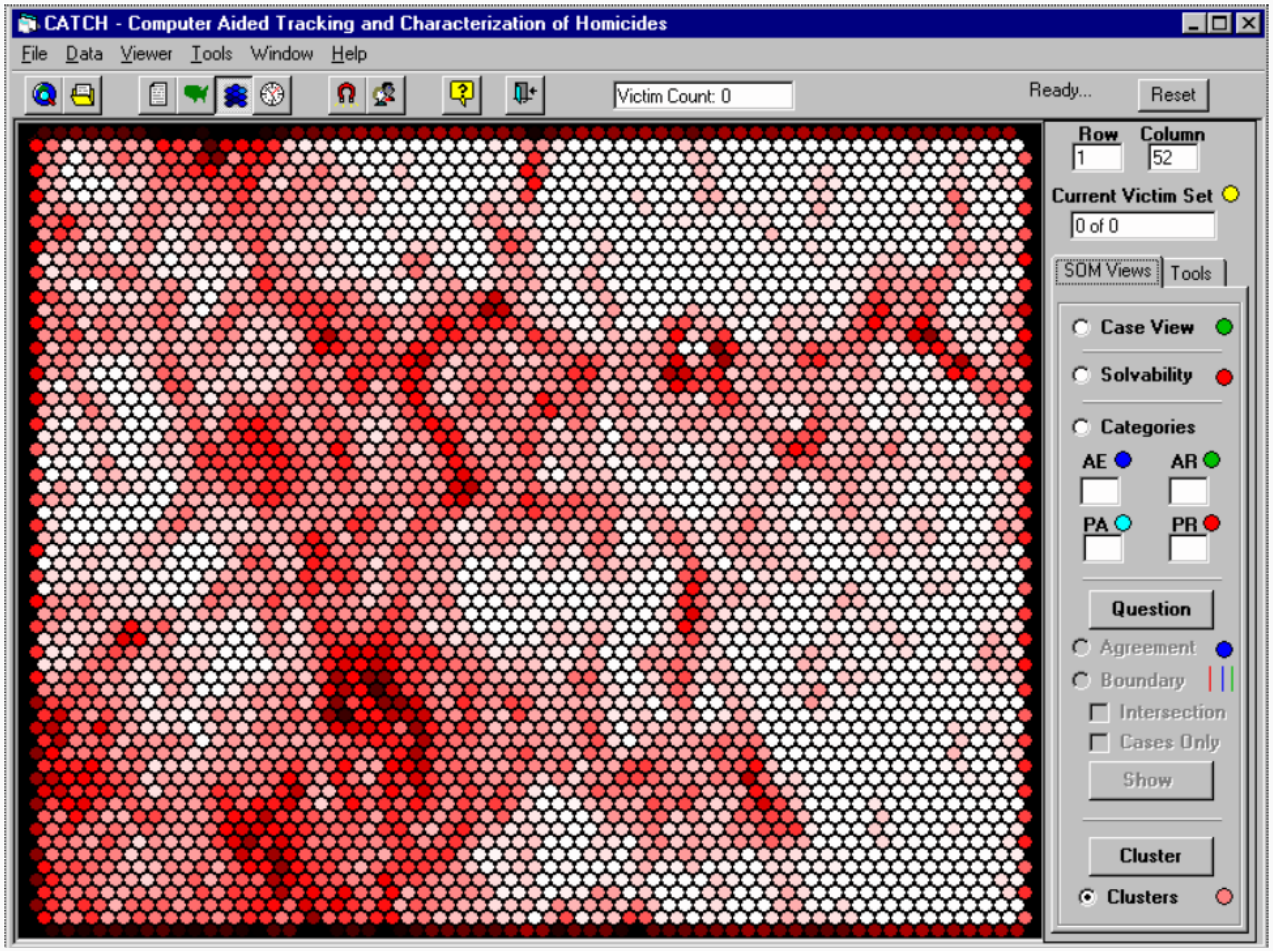

Figure 3: Ejemplo de uso de SOM en aplicaciones de perfilado. Tomado de [4].

Problemas y soluciones

Modelo 1: Problema

Tweet

> *"Really excited to add @plaidavenger to my* ***#deathlist*** *along with Italy and @Plaid_Obama after receiving that information.* ***#KillEveryone #ISIS****"*

Modelo 1: Predicción de etiquetas de Twitter

¿Que hacer?
Con un modelo de regresión lineal predecir los hashtags de los tweets.

"Really excited to add @plaidavenger to my deathlist along with Italy and @Plaid_Obama after receiving that information." $\Rightarrow$ **#deathlist, #KillEveryone, #ISIS**

Representación de palabras: Bag of Words

N es el tamaño del diccionario de términos D (e.g. $N = |D|$).

$$\text{word2idx} = \Big\{(t_i, i) : \forall i \in \{1, \ldots, N\}\Big\}$$

$$\text{idx2word} = \Big[t_1, \ldots, t_N\Big]$$

Representación de palabras en vectores para BoW
Para un termino individual su vector representativo se define como:

$$\boldsymbol{e}^{(i)} = [0, \ldots, 1, \ldots, 0] \leftarrow \text{posicion } i\text{–ésima}$$

$$\boldsymbol{e}^{(i)}, (t, i) \in \text{word2idx}$$

Para un documento d de términos, se calcula por cada termino que existen dentro del diccionario su vector representativo como:

$$\boldsymbol{s} = \sum_{(t,i) \in \text{word2idx}} \boldsymbol{e}^{(i)}, t \in d$$

Representación de palabras: TF–IDF

TF–IDF = Term Frequency – Inverse Document Frequency

Propósito
Darle mayor importancia a las palabras que ocurren con frecuencia intermedia en el documento d y en el corpus D.

$$\text{tf}(t, d) = \text{Frecuencia del termino (o n–grama) } t \text{ en el documento } d$$

$$\text{idf}(t, D) = \log\left(\frac{N}{|\{d \in D : t \in d\}|}\right); N = |D|$$

$$\text{tf-idf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D)$$

Regresión lineal

Para una vector de parámetros $\boldsymbol{\theta}$ y un vector de características $\boldsymbol{x}$, la regresión lineal se puede definir como:

$$\hat{y}(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \boldsymbol{x} = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

Donde $\hat{y}(\boldsymbol{x}, \boldsymbol{\theta}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$.

θ_0 se le conoce como el *bias* del modelo.

El objetivo es que para una salida esperada y se tenga la salida $\hat{y}$ con menor error por medio de ajustar los valores de $\boldsymbol{\theta}$. De forma que se quiere:

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}} |\hat{y}(\boldsymbol{x}, \boldsymbol{\theta}) - y|$$

Regresión logística $\sigma(x)$

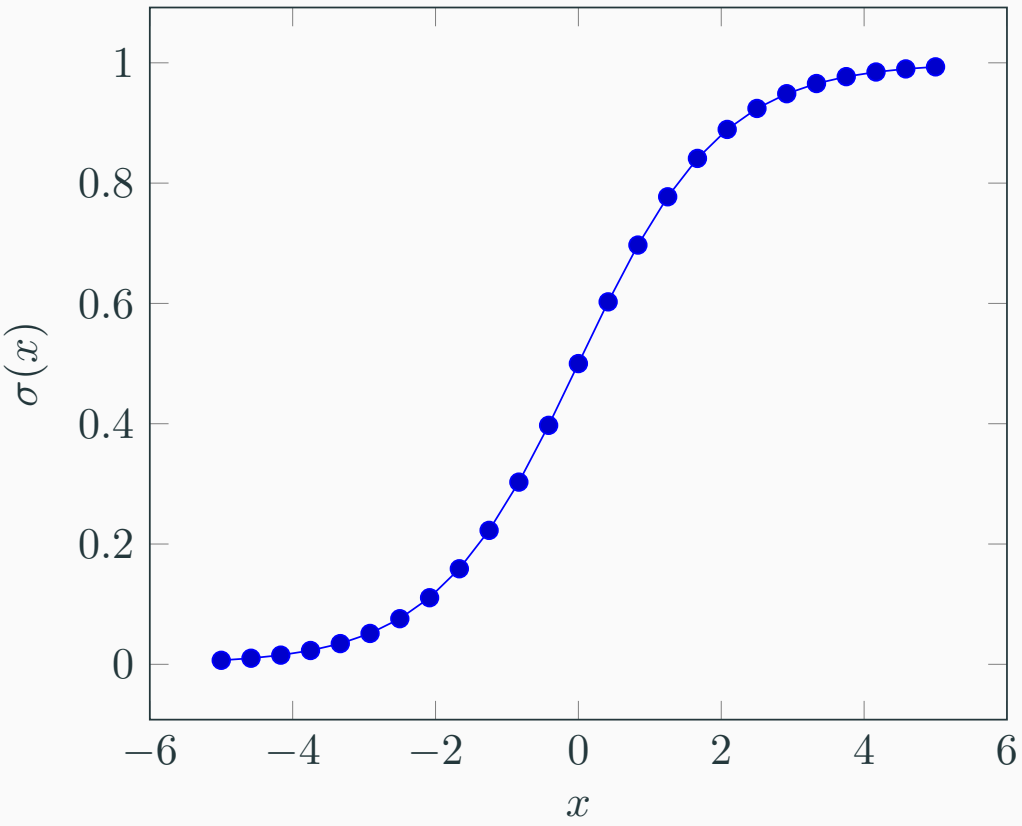

Figure 4: Gráfica de función sigmoide.

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

$$\sigma(x) : \mathbb{R} \to (0, 1)$$

Evita problemas de BIAS y OVERFITTING del modelo

One vs Rest

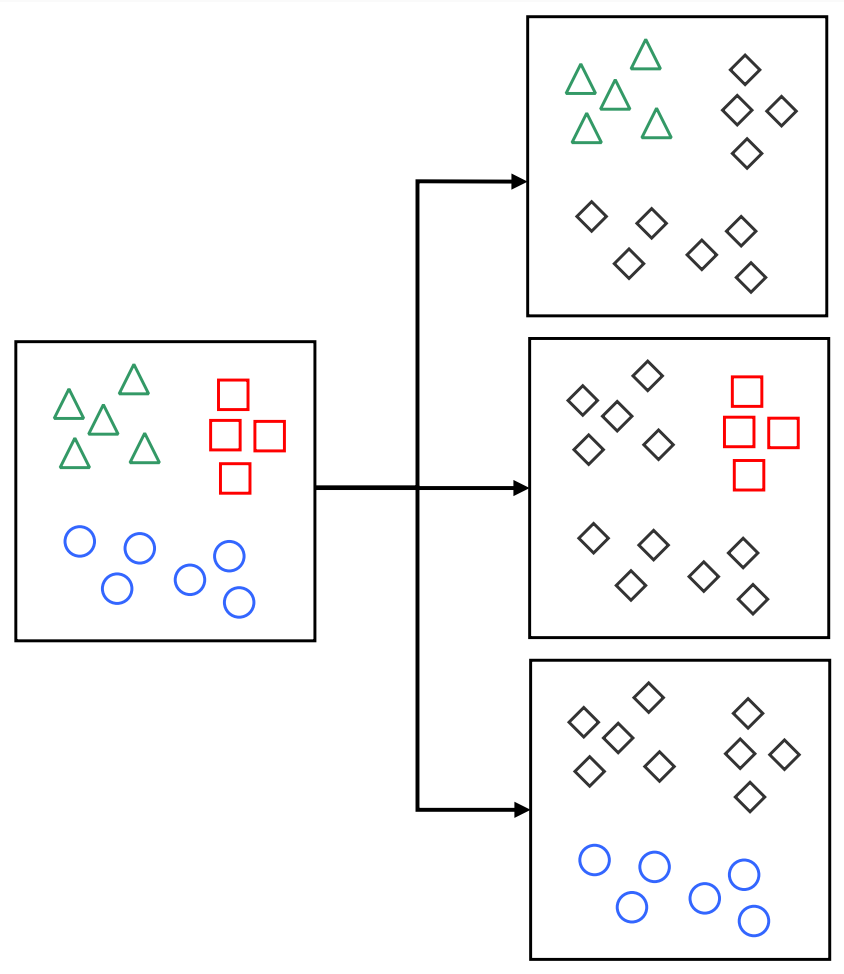

Figure 5: Algoritmo de One vs Rest.

Se entrenan C estimadores $\boldsymbol{\theta}_i$ para cada clase con algún algoritmo de optimización (ej. gradiente descendiente).

Se determina un estimador $c \in \{1, \ldots, C\}$, que se calcula como:

$$c = \arg\max_i \sigma(\boldsymbol{\theta}_i^\top \boldsymbol{x})$$

Predicción de hashtags

A partir de un diccionario previamente definido a entrenar el One vs Rest de forma:

$$\{(i, h)\}\,;\, h \in \text{hashtags}\,;\, i \in \{1, \ldots, C\}$$

De forma que se recupera el hashtag h correspondiente a partir de la clase estimada i por One vs Rest.

Modelo 2: Problema

¿De que y de quienes están hablando?

Tweet: @realDonaldTrump

> *"The **Democrats** new and pathetically untrue sound bite is that we are in a "Constitutional Crisis." They and their partner, the **Fake News Media**, are all told to say this as loud and as often as possible. They are a sad JOKE! We may have the strongest **Economy** in our history, best ..."*

Modelo 2: Reconocimiento de Named Entities con redes LSTM

Son redes neuronales recurrentes que son capaces de reconocer Named Entities.

Texto	Donald	Trump	es	presidente	de	Estados	Unidos
Etiqueta	B-PER	I-PER	O	O	O	B-ORG	I-ORG

Table 1: Ejemplo de reconocimiento de Named Entities.

Otro	O
Persona	PER
Ubicación	LOC
Organización	ORG
Misceláneo	MISC

Table 2: Categorías de Named Entities.

Redes neuronales recurrentes (RNN)

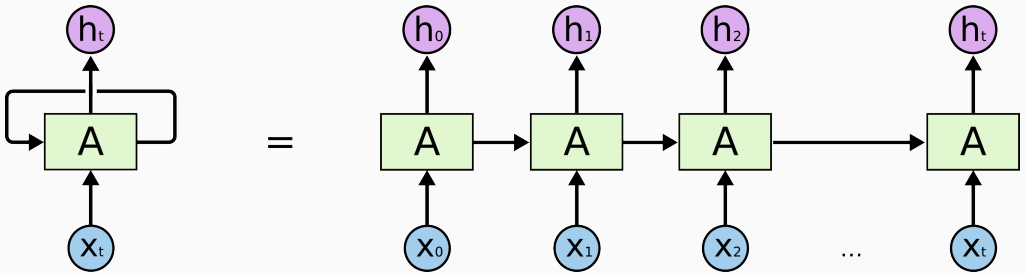

Figure 6: Red RNN simplificada. Tomado de [1].

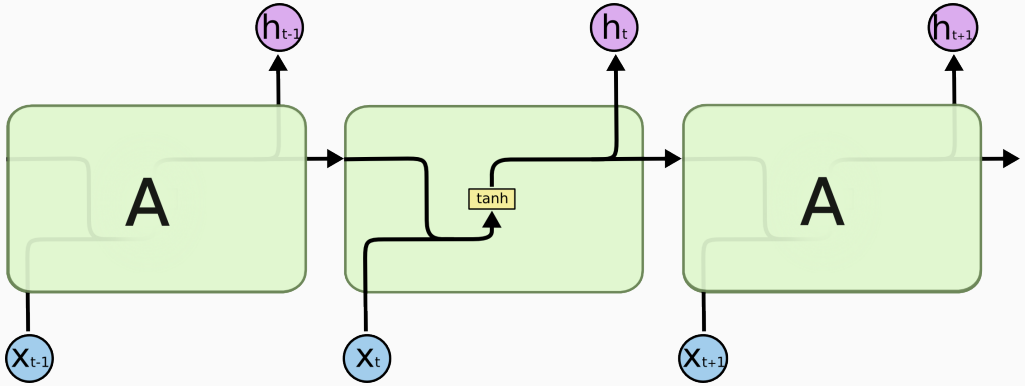

Figure 7: Arquitectura RNN clásica. Tomado de [1].

Redes Long Short Term Memory (LSTM)

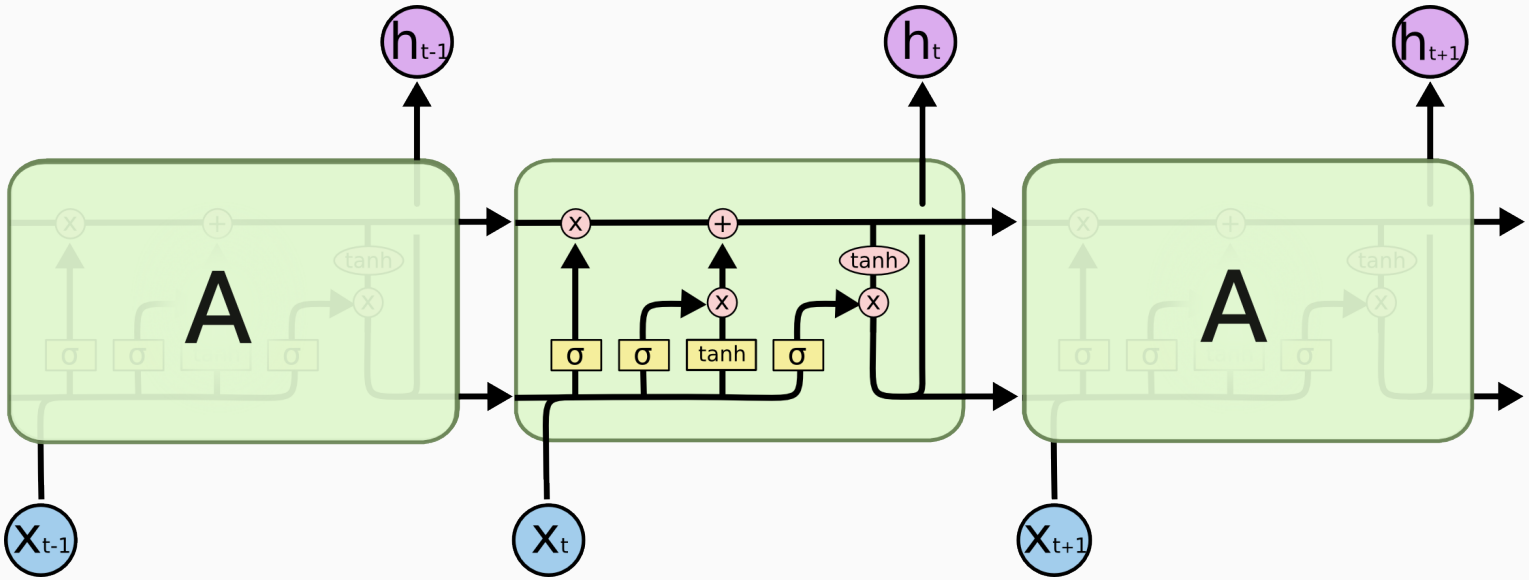

Figure 8: Arquitectura de red LSTM clásica. Tomado de [1].

Nota

Estas redes son solo *feedforward* (e.g. hacia adelante). Solo se basan en entradas pasadas.

Redes Bidirectional Long Short Term Memory (Bi-LSTM)

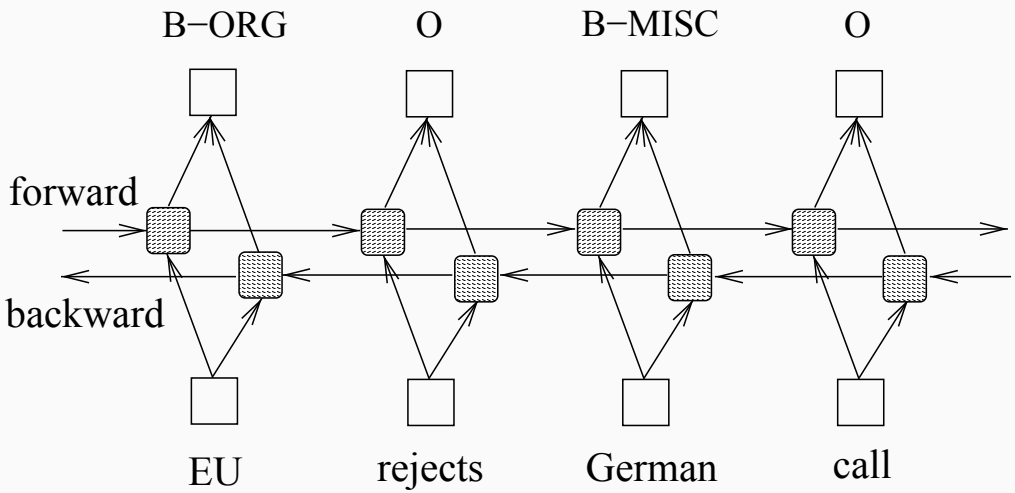

Figure 9: Etiquetado con una Bi-LSTM. Tomado de [3].

Nota

Estas redes son *feedforward* como *backward*. Se basan de entradas pasadas y futuras.

¿Que son los *embeddings*?

Son espacios de vectores n–dimensionales que se mapean según una palabra.

Tómese $\boldsymbol{p}_t$ como el vector que representa el termino t y a d como la distancia calculada entre los vectores (típicamente la distancia **coseno**).

Ejemplo

$d(\boldsymbol{p}_{\text{asombroso}}, \boldsymbol{p}_{\text{genial}})$ debería tener un valor bajo.

$d(\boldsymbol{p}_{\text{asombroso}}, \boldsymbol{p}_{\text{terrible}})$ debería tener un valor alto.

También se pueden representar varias palabras de un documento en un solo vector por medio de sumarlos. (i.e. $\sum_{t \in d} \boldsymbol{p}_t$).

Modelo 3: Búsqueda de tweets relacionados con *embeddings*

Es posible categorizar los k textos mas parecidos a una consulta q en base a su embedding con otros textos recopilados.

StarSpace
Genera embeddings en base a un dataset de entrenamiento.
Desarrollado por Facebook Research en 2017 [6].

Conclusiones y trabajo futuro

Conclusiones

- Se investigaron diferentes metodologías de NLP y Data Science para la tarea de perfilado de cibercriminales por medio de informacion de fuentes abiertas.
- Es necesario probar las metodologías propuestas con información obtenida de fuentes abiertas que este validada de forma que el entrenamiento de ellos sean efectivos en la tarea.

Trabajo futuro

- Implementación de los modelos 2 y 3 propuestos con propósito de ayudar al perfilamiento de cibercriminales.
- Recopilar datos pertinentes para el entrenamiento de los modelos propuestos.
- Adaptar y generalizar los modelos para el uso del lenguaje español.
- Implementar un modelo de recolección de información de redes sociales de cibercriminales de forma que sea mas fácil perfilarlos contra futuros.
- Realizar una visualización en dashboard de los algoritmos propuestos para ayudar al agente a realizar el perfilamiento.

Bibliografía i

[1] Understanding lstm networks.

[2] L. N. De Castro.
Fundamentals of natural computing: basic concepts, algorithms, and applications.
Chapman and Hall/CRC, 2006.

[3] Z. Huang, W. Xu, and K. Yu.
Bidirectional LSTM-CRF Models for Sequence Tagging.
2015.

[4] J. Mena.
Investigative Data Mining for Security and Criminal Detection.
Elsevier Science, 2003.

Bibliografía ii

[5] P.-N. Tan, M. Steinbach, and V. Kumar.
Introduction to Data Mining.
Addison Wesley, us ed edition, May 2005.

[6] L. Wu, A. Fisch, S. Chopra, K. Adams, A. Bordes, and J. Weston.
Starspace: Embed all the things!
arXiv preprint arXiv:1709.03856, 2017.