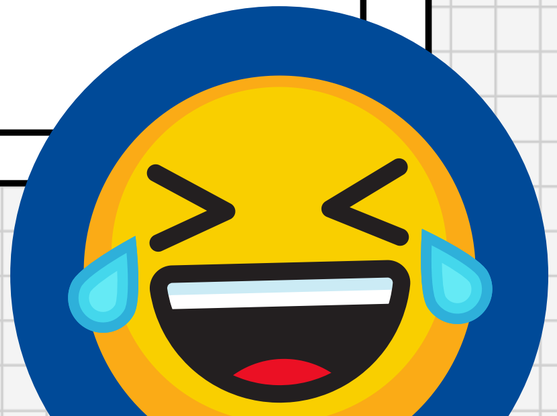
CC 5205- 2

30/05/22

HITO 2

Grupo N°1: Felipe Urrutia, Camilo Carvajal, José Saffie, Gianluca Musso, Matias Lopez



Motivación

"Procesamiento de Lenguaje Natural con Emojis"

Durante los últimos años hemos vivido un aumento considerable tanto en la cantidad y acceso a grandes cantidades de datos.

En este contexto, el procesamiento de lenguaje natural ha adquirido un rol protagónico, puesto que a traves de este, se puede predecir el valor sentimental de cierto conjunto de palabras.

"Muchisimas felicidaes mi amor 🥰 pd no me mates por el video @user"

Los Datos

Barbieri, F., Camacho-Collados, J., Ronzano, F., Espinosa Anke, L., Ballesteros, M., Basile, V., ... & Saggion, H. (2018). Semeval 2018 task 2: Multilingual emoji prediction. In 12th International Workshop on Semantic Evaluation (SemEval 2018) (pp. 24-33). Association for Computational Linguistics

Predicción de emoji multilingual

Inglés

❤️😍😂💕🔥😊😎✨💙😘
📷🇺🇸☀️💜😉💯😁🎄📸😜

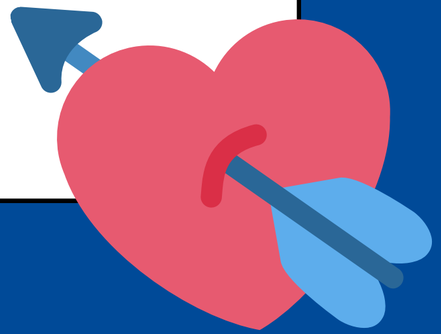
Español

❤️😍😂💕😊😘💪😉👌🇪🇸
😎💙💜😜💞✨🎶💘😁

	Trial	Training	Test
Inglés	50,000	500,000	50,000
Español	10,000	100,000	10,000

Mejoras Hito 1

- Se eliminó la pregunta 4, debido a su dificultad y al tiempo de realización.
- Se realizó la exploración de datos del dataset en español.



¿Es posible utilizar un modelo predictivo que reciba el texto de un tweet y prediga su emoji asociado?

Esta es la principal pregunta a responder en el dataset y fue usada para la competencia SEMEVAL de procesamiento de lenguaje natural. Para responderla se usarán clasificadores de dos tipos:

- Naïve-Bayes: este clasificador nos permite usar la proporción de palabras por clase para definir una probabilidad de pertenencia a estas. Usando la suposición "naïve" de que la ocurrencia de palabras en un tweet son independientes, se genera una distribución de probabilidad sobre cada emoji. Se analizarán estas probabilidades como si fuesen representaciones; intrepretaremos los tokens con mayor probabilidad para cada clase; y realizaremos un grid-search para seleccionar los mejores parámetros.

- Transformers: esta arquitectura de red neuronal ha sido clave en el auge del NLP. Aprovecharemos la existencia de modelos pre-entrenados para utilizarlos en esta tarea. Compararemos los resultados de algunos modelos con aquellos de Naive-Bayes y además visualizaremos capas de atención.

¿Podemos utilizar representaciones vectoriales apropiadas de los tweets y encontrar algoritmos clustering capaces de relacionar aquellos tweets asociados a un mismo emoji?

Posterior al preprocesamiento de los datos realizado en el hito 1, se busca relacionar los tweets asociados a un mismo emoji a partir de algoritmos de clustering, tales como : k-means, Jerárquico aglomerativo, dbscan, optics y Mixturas Gaussianas (Tabla 2). Esto nos permitirá encontrar el método mas apropiado para predecir y encontrar patrones entre tweets asociados a un mismo emoji.

Para poder aplicar los algoritmos de clustering, se deberán representar vectorialmente cada tweet, tanto en ingles como en español, a través de técnicas como: bag-of-word, tf-idf , word2vec, BETO y BERTweet (Tabla 1).

Finalmente se evaluará cada algoritmo de forma cuantitativa y cualitativa, ocupando métricas como Rand index, Mutual Information, Homogeneity, Completeness y V-measure para la parte cuantitativa (Tabla 3). Por otra parte, para la evaluación cualitativa, se debe reducir la dimensionalidad de los vectores (2 - 3 dimensiones), a partir de los métodos PCA, tSNE y UMAP.

Tabla 1. *Representación del tweet*

Notación	Método	US	ES	Libreria
BOW	bag-of-words	✔	✔	sklearn
TFIDF	term-frecuency invert-document-frecuency	✔	✔	sklearn
W2V	Word2vec	✔	✔	gensim
BETO	BETO: Spanish BERT		✔	🤗
BERTweet	BERTweet: A pre-trained language model for English Tweets	✔		🤗

Tabla 2. *Algoritmos de clustering*

Notación	Método	param: #(clusters)	Libreria
kMeans	k-means	✔	sklearn
Agg	Agglomerative Hierarchical	✔	sklearn
DBSCAN	Density-based spatial clustering of applications with noise		sklearn
OPTICS	Ordering points to identify the clustering structure		sklearn
GM	Gaussian Mixture	✔	sklearn

Tabla 3. *Métricas de evaluación para clustering. El simbolo 🔒 indica que es necesaria dicha información para calcular la metrica*

Notación	Método	Rango	Ground truth (Etiquetas)	kMeans	Agg	DBSCAN	OPTICS	GM	Libreria
Corr	Correlación (Pearson)	[-1, 1]^n	✔	✔	✔	✔	✔	✔	🐼
SC	Silhouette	[-1, 1]	✔	✔	✔	✔	✔	✔	sklearn
Rand	Rand index	[0, 1]	🔒	✔	✔	✔	✔	✔	sklearn
NMI	Mutual Information	[0, 1]	🔒	✔	✔	✔	✔	✔	sklearn
Hom	Homogeneity	[0, 1]	🔒	✔	✔	✔	✔	✔	sklearn
Comp	Completeness	[0, 1]	🔒	✔	✔	✔	✔	✔	sklearn
V	V-measure	[0, 1]	🔒	✔	✔	✔	✔	✔	sklearn

¿Se puede asociar a una palabra, un índice de probabilidad de estar asociado a un emoji en específico?

Se proponen dos formas para encontrar una probabilidad asociada a la palabra, la primera consiste en utilizar el mismo de método de clasificación que se utilizo para la primera pregunta, es decir, realizando el mismo pre-procesamiento, usando naive bayes y luego obteniendo el vocabulario de todas la palabras, se puede buscar que probabilidad tiene asociada cada palabra para cada emoji.

La segunda forma consiste en realizar el mismo pre-procesamiento pero esta vez haciendo una regresión lineal utilizando la librería de pandas, con esto creando una tabla completamente de carácter binario donde cada fila representa un tweet, los primeros atributos van a decir que a que emoji corresponde y el resto de atributos van a ser todas las palabras que están presentes en todos los tweets. Con esto se pueden hacer 2 regresiones diferentes una considerando todas las palabras o considerando solo una palabra, el caso general que así: $Y_{\text{emoji}} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \varepsilon$

Donde Y corresponde al emoji que se quiere predecir, los X son un variables binarias que dicen si esta una palabra en especifico presente en el tweet y el beta asociado es la probabilidad que aporta cada palabra para predecir el emoji Y. Si bien el método de regresión lineal puede resultar poco eficiente, es otra forma de asociar probabilidades a cada palabra por cada emoji. Para determinar la calidad de este predictor se puede testear si en cada tweet, el Y con mayor probabilidad sea efectivamente el emoji en cuestión del tweet.

¿Un Hashtag es más útil a la hora de entrenar un modelo para predecir un emoji ?

Para poder responder esta pregunta, se consideraran solo los tweets que tienen hashtags y se tendrán 3 dataframe, uno con el tweet original, otro con los hashtags y el ultimo con el contenido del tweet sin hashtags.

Ahora bien, para poder aplicar cualquier algoritmo se deben eliminar números, puntuación(.,!,#) y dejar todo el contenido del hashtag en minúscula. Por otra parte, de ser necesario se realizará un reajuste en el balance de clases mediante subsampling.

Luego, se ocupará el método de KNN, puesto que este es usado frecuentemente para búsquedas semánticas, asociando k puntos de datos más cercanos del dataset train a el punto de consulta (puede ser con distancia euclidiana) y además se utilizará el mejor clasificador que nos entregue la pregunta 1, esto para poder hacer una comparación entre clasificadores y obtener los mejores resultados.

Finalmente se medirá la efectividad de cada clasificado con las métricas F1 score y sensibilidad, puesto que existe un desbalance entre clases (cantidad de cada emoji), haciendo que accuracy no sea la mejor métrica en este caso (de igual forma se espera un porcentaje sobre el 70%).

¿Qué emojis son más fáciles de predecir, cómo y por qué?

Search

Luego del preprocesamiento propuesto en el hito 1, se propondrán dos métodos para resolver esta pregunta:

La primera opción es analizar los inputs que tienen más peso en el modelo. Para esto se podrían utilizar librerías cómo SHAP o LIME, que permiten encontrar qué emoji es más fácil o más difícil de predecir, a través de cuanto contribuye cada entrada al modelo.

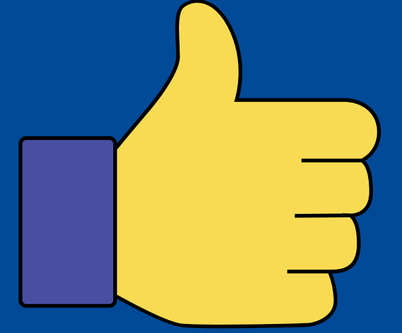
Una segunda opción sería realizar una reducción de dimensionalidad haciendo un PCA (Principal Component Analysis), puesto que de esta forma podemos definir los componentes más importantes del modelo. Para poder medir el nivel de relevancia de cada componente, se utiliza la varianza de cada uno, dando como resultado que el emoji que menos aparece será muy importante.

Dado esto, un emoji que tiene una frecuencia menor en la base de datos, será más fácil de predecir, ya que su significado generalmente es interpretado de una única forma (un ejemplo puede ser un emoji de calavera, el cual está relacionado principalmente a la muerte). Por otra parte, un emoji que tiene una frecuencia mucho mayor, cómo una cara sonriendo, será mucho más complejo de predecir, debido a que se le atribuyen diferentes interpretaciones dependiendo del contexto, ya sea en un tweet celebrando por algún suceso “x” o incluso por alguna situación sarcástica.

Save

Cancel

Resultados Clasificador Bayes navie



Grid Search

Para realizar una mejor predicción se eliminaron tokens los cuales se repitieran muy poco en el entrenamiento, es por eso que mediante un grid-search se compararon varias iteraciones del entrenamiento, donde en cada una se escogió un cantidad de repeticiones mínimas para no eliminar el token y un alpha que denota parámetros de “smothness“ para el clasificador

Se utilizo el grid con la métrica de f1-macro ya que esta permite evaluar el algoritmo, sin que el desbalance de los emojis genere un sesgo al entrenar este

Top de palabras por cada emoji

El clasificador bayes naive asocia cada a palabra del set de entrenamiento a una probabilidad según que emoji se este prediciendo, además podemos obtener la lista de palabra con los que se realizo el entrenamiento del algoritmo, esto considerando también las palabras que se desecharon después del grid-search que se realizo. Luego con python se puede realizar un búsqueda y encontrar las palabras que tengan asociada la probabilidad mas alta según cada emoji, y de esta forma crear un top de palabras por cada emoji.

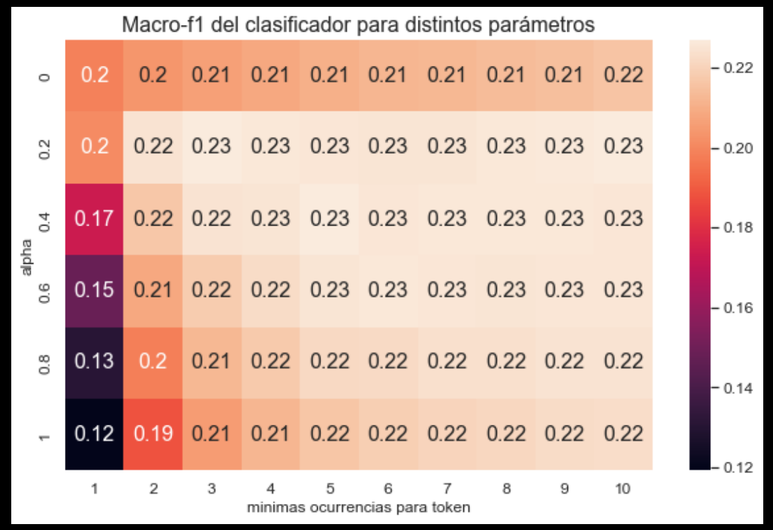
UMAP de vectores de probabilidad de tokens

Para poder utilizar este método de visualización es importante primero reducir la dimensionalidad a 2 para poder trabajar con los datos, luego con esto obtenemos vectores de probabilidad bidimensionales, para poder diferenciar cada emoji, estos se colorean según un color en especifico, además se atribuyen los tokens a aquellas clases donde tienen mayor probabilidad para facilitar el análisis, también se muestra con diferentes tamaños para tener una clara representación de que tokens tiene una mayor valor probabilistico a la hora de querer predecir una emoji.

Métricas de evaluación sin eliminar palabras poco frecuentes

	precision	recall	f1-score	support
🤍	0.35	0.58	0.44	10798
😍	0.25	0.25	0.25	4830
📸	0.16	0.16	0.16	1432
us	0.47	0.50	0.48	1949
☀️	0.25	0.43	0.32	1265
💜	0.32	0.05	0.08	1114
😉	0.12	0.04	0.06	1306
💯	0.27	0.14	0.19	1244
😁	0.14	0.03	0.05	1153
🎄	0.60	0.60	0.60	1545
📸	0.29	0.10	0.15	2417
😜	0.04	0.01	0.01	1010
😂	0.30	0.52	0.38	4534
💞	0.19	0.05	0.08	2605
🔥	0.45	0.47	0.46	3716
😊	0.09	0.06	0.07	1613
😎	0.16	0.11	0.13	1996
✨	0.29	0.18	0.22	2749
💙	0.22	0.07	0.10	1549
😘	0.16	0.05	0.08	1175
accuracy			0.32	50000
macro avg	0.26	0.22	0.22	50000
weighted avg	0.29	0.32	0.28	50000

Grid Search para utilizando macro-f1



Métricas de evaluación cuando no eliminan palabras que se repitan menos de x y usando alpha = y

	precision	recall	f1-score	support
🤍	0.38	0.48	0.42	10798
😍	0.26	0.24	0.25	4830
📸	0.14	0.17	0.16	1432
us	0.42	0.52	0.46	1949
☀️	0.23	0.49	0.31	1265
💜	0.23	0.06	0.10	1114
😉	0.10	0.08	0.09	1306
💯	0.21	0.20	0.20	1244
😁	0.10	0.06	0.07	1153
🎄	0.57	0.64	0.60	1545
📸	0.26	0.15	0.19	2417
😜	0.07	0.03	0.04	1010
😂	0.33	0.47	0.39	4534
💞	0.18	0.08	0.11	2605
🔥	0.45	0.45	0.45	3716
😊	0.09	0.07	0.08	1613
😎	0.15	0.12	0.13	1996
✨	0.27	0.21	0.23	2749
💙	0.19	0.10	0.13	1549
😘	0.13	0.10	0.11	1175
accuracy			0.31	50000
macro avg	0.24	0.24	0.23	50000
weighted avg	0.28	0.31	0.29	50000

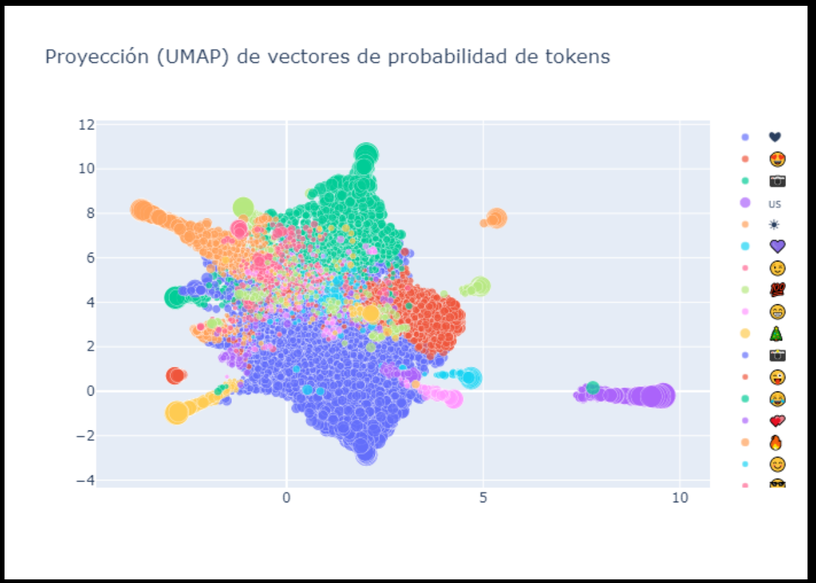
Ejemplos de Top de palabras por emoji

📸
```
(['mugshot', 'mag', 'opus', 'bricks', 'banshee'],
 array([0.42634416, 0.45683482, 0.64613312, 0.51130967, 0.48844705]))
```

😍
```
(['asada', 'eatgood', 'gorg', 'beignet', 'swooning'],
 array([0.5990263 , 0.5996593 , 0.62417838, 0.6351927 , 0.6352712 ]))
```

✨
```
(['peterpan', 'pixie', 'getonshimmur', 'mmxvi', 'sparkle'],
 array([0.53351226, 0.56177346, 0.74150652, 0.57578178, 0.6144488 ]))
```

UMAP de vectores de probabilidad de tokens



Muchas Gracias!