# Proyecto Final

HAHA (Humor Analysis based on Human Annotation) es una competencia de procesamiento del lenguaje natural en la que el problema consiste en clasificar textos cortos del español (obtenidos de Twitter) como humorístico (clase 1) o no humorístico (clase 0) usando cualquier técnica. El dataset contiene más de 10 mil ejemplos de entrenamiento con sus respectivas clases y un conjunto de datos para test sin clases para su predicción.
En este caso se usarán redes neuronales basadas en propagación hacia adelante y experimentación para determinar los hiper parámetros con mejores resultados.

# Experimentación

La estrategia de experimentación consistió en una exploración limitada de los diferentes variables en los modelos, para definir el hiper parámetro con un valor específico, una variable a la vez. Se implementó un ciclo principal para todos los valores elegidos de la variable, el entrenamiento luego se hizo con K-folds eligiendo como ganador aquel modelo que tenga mejor F1-score en promedio entonados los K-folds de su grupo.

## Arquitectura de capas completamente conectadas

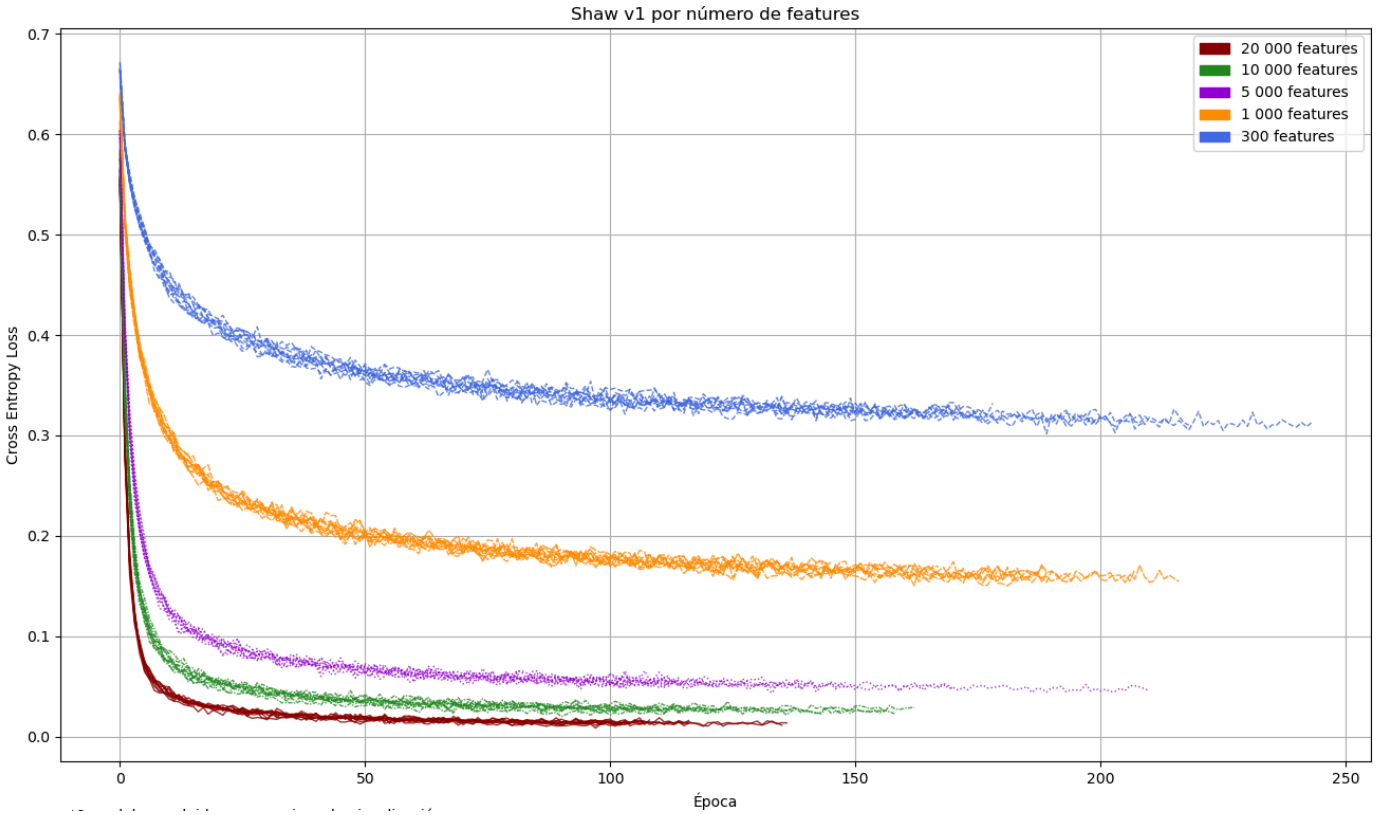
El modelo base implementado sobre el cual se comenzó a experimentar consiste en las siguientes características:

- Learning rate: 0.001
- Batch size: 32
- Vectorización: TF-IDF (normalización)
- Capas ocultas: 3 (64 neuronas cada una)
- Dropout: 20% en la primera capa, 50% en las capas ocultas
- Optimización: Adam
- Error: Cross Entropy Loss
- Activación: ReLu
- Neuronas salida: 2 (Codificación One-Hot)
- Early stop: Si el error es menor a 0.0001 o si no ha habido mejora de 0.000001 respecto a las últimas 15 épocas
- Balanceo de clases: SMOTE
- K-folds: 10

El primer parámetro con el que se experimentó fue con el número máximo de features, se evaluó con 5 posibilidades: 20 mil features, 10 mil features, 5 mil features, mil features y 300 features. Los resultados promediados de los K-folds se muestran en la tabla.

| **Max features** | **20 000** | **10 000** | **5 000** | **1 000** | **300** |
|---|---|---|---|---|---|
| **F1-score promedio (10 folds)** | 0.76351521 | 0.76445018 | 0.76814913 | 0.75255446, | 0.72225481 |

Como se observa en la gráfica mayor cantidad de features generalmente hace una convergencia en un error menor, dados los promedios de F1-score y la cantidad de tiempo que toma entrenar modelos con mayor cantidad de features, para los siguientes experimentos se modificó el máximo de features a 5 mil.

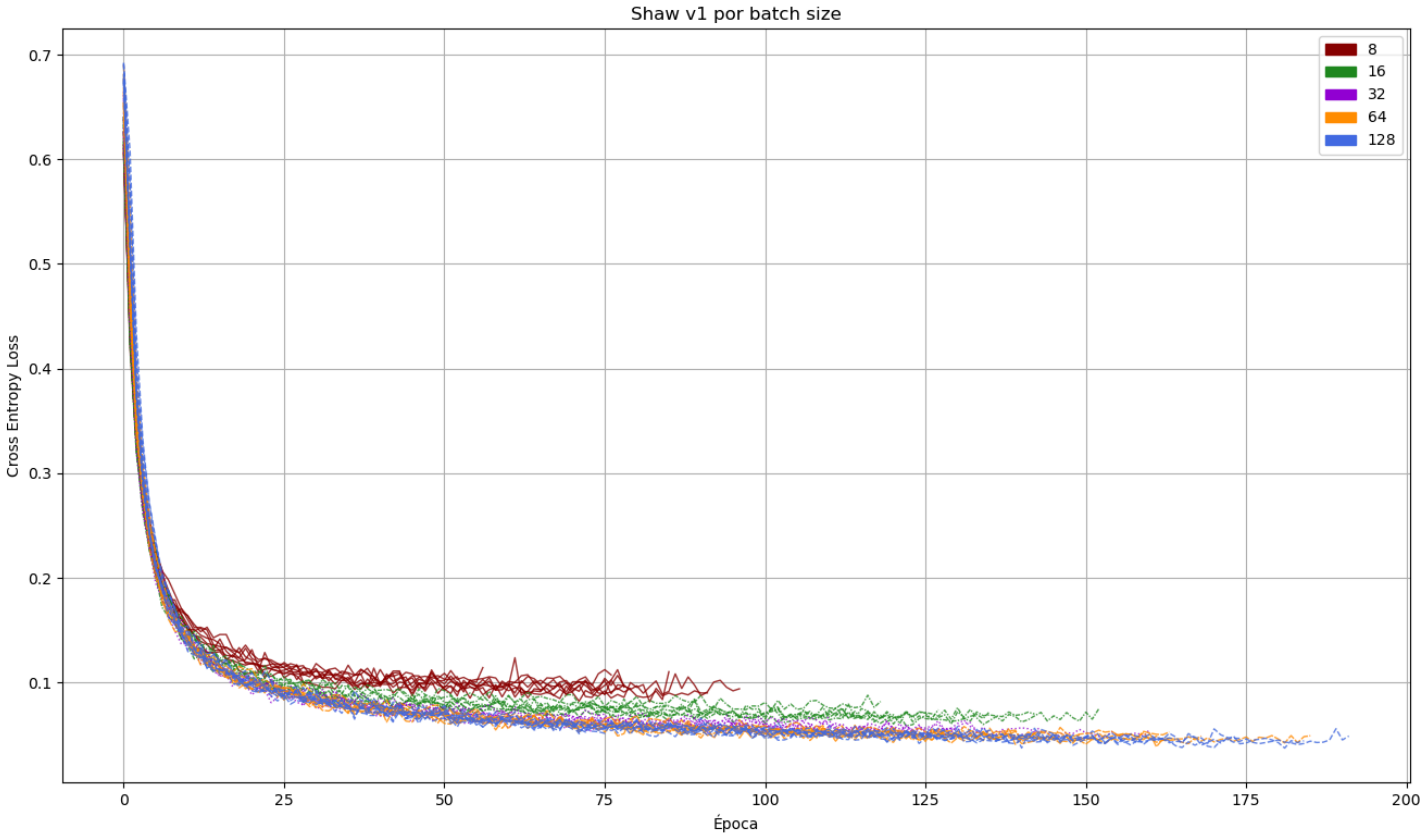



A continuación se re-evaluaron las variables del número capas y tamaño de batch, pues tanto la arquitectura inicial propuesta como el batch fue solo una base de la que partir. Los resultados se resumen en la siguiente tabla

| **Número de capas (64 neuronas c/u)** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| **F1-score promedio (10 folds)** | 0.7411309 | 0.76788326 | 0.76847381 | 0.7679649 | 0.76957909 |

| **Batch size** | **8** | **16** | **32** | **64** | **128** |
|---|---|---|---|---|---|
| **F1-score promedio (10 folds)** | 0.76470965 | 0.76236847 | 0.76683344 | 0.76227912 | 0.76900614 |

Estos nuevos valores fueron utilizados en los siguientes experimentos. De igual forma que con el número de features, con el batch size se observa que entre mayor sea mas bajo es el valor de la convergencia del error.




Tras definir los hiper parámetros para TF-IDF se volvieron a aplicar pero con una representación de textos de Word Embeddings. Se experimentó cuál modelo de variación del español funcionó mejor.

| Variación del español (fasttext) | México | España | Español (general) |
|---|---|---|---|
| **F1-score promedio (10 folds)** | 0.75911244 | 0.75887624 | 0.76578043 |

Como se observa por sí solo, word embeddings empeora el rendimiento general del modelo, por lo que se exploraron nuevas arquitecturas.

## Arquitectura convolucional.

Se agregó una capa convolucional (1D) a la arquitectura anterior, dicha procesaba los datos antes que la primera capa completamente conectada. Los resultados obtenidos fueron desalentadores ya que la mayoría de los modelos directamente no convergían en ningún valor y por tanto predecían de manera completamente errática. Por ello, esta arquitectura fue desechada.

## Arquitectura de fusión simple

Se construyó una arquitectura que fusionara los métodos de vectorización antes utilizados, esta red neuronal tiene 2 entradas correspondientes a las 2 representaciones vectoriales del mismo texto (TF-IDF y word embeddings. La salida de cada una de las capas corresponde a la mitad de las neuronas de la primera capa oculta, de manera que concatenando ambas salidas se hacian pasar los datos por la misma arquitectura de 5 capas descrita anteriormente, obteniendo así un modelo de fusión simple.
Tras construir la nueva arquitectura se probó con los mismos parámetros ya definidos pero el desempeño del modelo fue aún peor que antes, por lo que se volvió a evaluar la cantidad de features máxima de TF-ID, pues

el desbalance de features entre las 2 entradas posiblemente afecta al desempeño del modelo. Los resultados se muestran en la siguiente tabla.

| Max features | 5 000 | 2 500 | 1 000 | 500 | 300 |
|---|---|---|---|---|---|
| F1-score promedio (10 folds) | 0.76563751 | 0.76359202 | 0.76142486 | 0.7618873 | 0.76853873 |

Del mismo modo la cantidad de neuronas de las capas completamente conectadas fueron reevaluadas. La capa de fusión es la misma que la primera capa oculta del modelo anterior.

| Neuronas de la capa oculta #1 (capa de fusión) | 64 | 128 | 256 |
|---|---|---|---|
| F1-score promedio (10 folds) | 0.7707721 | 0.77623175 | 0.78407343 |

| Neuronas de la oculta capa #5 (capa de salida) | 256 | 128 | 64 | 32 | 16 |
|---|---|---|---|---|---|
| F1-score promedio (10 folds) | 0.78042249 | 0.78143724 | 0.78015475 | 0.78745798 | 0.78232972 |

| Neuronas de la oculta capa #2 (capa de salida) | 256 | 128 | 64 | 32 | 16 |
|---|---|---|---|---|---|
| F1-score promedio (10 folds) | 0.78337286 | 0.77953555 | 0.78745797 | 0.77746154 | 0.78147564 |

| Neuronas de la oculta capa #3 (capa de salida) | 256 | 128 | 64 | 32 | 16 |
|---|---|---|---|---|---|
| F1-score promedio (10 folds) | 0.79026219 | 0.78320559 | 0.78745797 | 0.78204189 | 0.78426785 |

# Shaw v3.5

El mejor modelo obtenido cuenta con las siguientes características.

- Learning rate: 0.001
- Batch size: 128
- Épocas: 500
- Vectorización
    - TF-IDF
        - Normalización del texto
        - 300 features
    - Word Embeddings (300 dimensiones)
- Arquitectura
    - 2 capas de entrada (Dropout p=0.2)
        - TF-IDF (128 neuronas)

        - Word Embeddings (128 neuronas)
    - 5 capas ocultas (Dropout p=0.5)
        - capa de fusión (256 neuronas)
        - capa oculta (64 neuronas)
        - capa oculta (256 neuronas)
        - capa oculta (64 neuronas)
        - capa oculta (32 neuronas)
    - 2 Salidas (2 clases en codificación One Hot)
- Optimización: Adam
- Error: Cross Entropy Loss
- Activación: ReLu
- Entrenamiento
    - Early stop
        - Si el error es menor a 0.0001
        - Si no ha habido mejora de 0.000001 respecto a las últimas 15 épocas
    - Balanceo de clases: SMOTE
    - Dataset de entrenamiento completo (sin partición de test ni evaluación)
- Random State: 45

## Comparación

La arquitectura desarrollada (Shaw v3.5) supera al modelo base de SVM + TF-IDF en F1-score, Presicion y Accuracy. Pero sigue siendo peor que el ganador del concurso del año anterior.

| Descripción del Modelo | Precisión | Recall | F1 | Accuracy |
|---|---|---|---|---|
| [CURSO RNA 2024] Arq_01_BA_02_g_v2 | 0.8004 | 0.8006 | 0.8005 | 0.8146 |
| MLP_P2_TOP1_V1 | 0.8039 | 0.7773 | 0.7866 | 0.8096 |
| Torch_NC_T1_V2_15000_2.5 | 0.7861 | 0.787 | 0.7866 | 0.8014 |
| Flora_Sirenix_v22 ((5, [6, 7]), (21, 69, 261), 5, 21, 105, 7) | 0.7898 | 0.781 | 0.7848 | 0.803 |
| Flora_Sirenix_v21 ((5, [6, 7]), (21, 69, 261), 5, 21, 105, 7) | 0.7891 | 0.7806 | 0.7843 | 0.8025 |
| Shaw v3.5 try4 | 0.811 | 0.7718 | 0.7838 | 0.8104 |
| MLP_P2_TOP2_V4 | 0.7969 | 0.7752 | 0.7832 | 0.8054 |
| Shaw v3.5 try2 | 0.8052 | 0.772 | 0.7828 | 0.808 |
| Shaw v3.2 | 0.7842 | 0.7812 | 0.7826 | 0.7991 |
| Flora_Sirenix_v20 ((5, [6, 7]), (21, 69, 261), 5, 21, 105, 7) | 0.7856 | 0.7771 | 0.7808 | 0.7993 |
| Torch_P_T1 | 0.7919 | 0.7731 | 0.7802 | 0.802 |
| Torch_P_T3 | 0.8184 | 0.7656 | 0.7797 | 0.81 |
| BASELINE: SVM + TF-IDF | 0.7914 | 0.7724 | 0.7796 | 0.8014 |

# Conclusiones

El diseño de arquitecturas de redes neuronales es un proceso complejo pues varios hiper parámetros dependen unos de otros, haciendo que sea imposible determinar uno solo en un valor fijo que de un mejor rendimiento. Esto es comprobable ya que al evaluar posibilidades una a una durante la experimentación, el modelo empeoró en algunos momentos dados.

La fusión de TF-IDF y word embeddings mejoró el modelo posiblemente gracias a la redundancia que aporta este método, aún si la gran mayoría de información de los vectores TF-IDF era limitada a 300 features, posiblemente porque balancea la importancia de los 2 métodos por igual.

Las capas de convoluciones parecen ser detrimentales para la clasificación de textos, posiblemente porque el mezclar información desorganizadamente oculta el significado original de los vectores, sin embargos es necesaria más experimentación con los parámetros de las capas convolucionales para asegurar que este es el caso; esta técnica es útil para otro tipo de problemas de clasificación, como aquellos que involucran el reconocimiento de imágenes, pues son vectores 2D naturales.