Momento de Retroalimentación: Módulo 2

Implementación de un modelo de deep learning

Reporte sobre el desempeño del modelo.

Nombre y Matrícula:

Ulises Orlando Carrizalez Lerín A01027715

TC3007C.501 - Inteligencia artificial avanzada para la ciencia de datos II (Gpo 501)

Benjamín Valdés Aguirre

Date:

08/11/2025

Índice

1. Introducción.....3

2. Dataset.....3

3. Redes Neuronales Recurrentes y GRU.....4

4. Modelo.....4

6. Resultados del modelo.....6

7. Mejoras y resultados finales.....7

8. Predicciones con el modelo.....10

9. Conclusión.....10

1. Introducción

Este proyecto tiene como objetivo desarrollar un modelo de machine learning capaz de identificar las emociones expresadas en textos breves, utilizando una red neuronal recurrente (RNN) de tipo Gated Recurrent Unit (GRU). Para ello, se propone un enfoque que procese secuencias de texto de manera eficiente, capturando las dependencias temporales entre palabras con el fin de lograr una clasificación precisa de las emociones presentes en los datos.

2. Dataset

Este proyecto hace uso del conjunto de datos de clasificación de emociones "Emotions Dataset" de Kaggle, este está diseñado para facilitar la investigación y la experimentación en el campo del procesamiento del lenguaje natural y el análisis de emociones. Contiene una colección diversa de muestras de texto, cada una etiquetada con la emoción correspondiente que transmite.

Las emociones del dataset se clasifican en seis categorías:

- tristeza (sadness)
- alegría (joy)
- amor (love)
- enojo (anger)
- miedo (fear)
- sorpresa (surprise)

Link al Dataset: https://www.kaggle.com/datasets/bhavikjikadara/emotions-dataset/data

3. Redes Neuronales Recurrentes y GRU

Una Red Neuronal Recurrente (RNN) es un tipo de arquitectura de red diseñada para procesar datos secuenciales, como texto o series temporales. A diferencia de las redes neuronales tradicionales, las RNN poseen conexiones que permiten mantener información de pasos anteriores, lo que les permite recordar el contexto de la información anterior y utilizarlo para predecir elementos en la secuencia. Sin embargo, las RNN convencionales suelen presentar problemas de vanishing lo que dificulta su entrenamiento con secuencias largas.

Las RNN de tipo GRU (Gated Recurrent Unit) introducen mecanismos de compuertas que controlan el flujo de información dentro de la red. Estas compuertas permiten decidir qué información conservar y cuál olvidar, mejorando la capacidad del modelo para capturar dependencias a largo plazo sin incrementar significativamente la complejidad computacional. En comparación con otras variantes como las LSTM, las GRU resultan más ligeras y rápidas de entrenar, manteniendo un desempeño competitivo en tareas de análisis de texto.

4. Modelo

Debido a las razones expuestas anteriormente, se decidió emplear una RNN de tipo GRU para abordar el problema de análisis de sentimientos entre las diferentes clases. Este tipo de arquitectura permite capturar las dependencias temporales presentes en las secuencias de texto de manera eficiente, facilitando la identificación de patrones emocionales. Para la implementación del modelo se utilizó el framework PyTorch, el cual ofrece una estructura flexible y un conjunto de herramientas optimizadas para el desarrollo y entrenamiento de modelos de deep learning.

En la fase de preprocesamiento, se generó un embedding de 100 dimensiones. Posteriormente, se realizó la generación del vocabulario, la tokenización y la creación de secuencias necesarias para la entrada de la red neuronal. Con base en estos datos, se construyeron los objetos Dataset y, posteriormente, los DataLoader, utilizados para entrenar el modelo.

Como se mencionó anteriormente, el modelo está basado en una arquitectura GRU, cuya arquitectura consiste en las siguientes etapas:

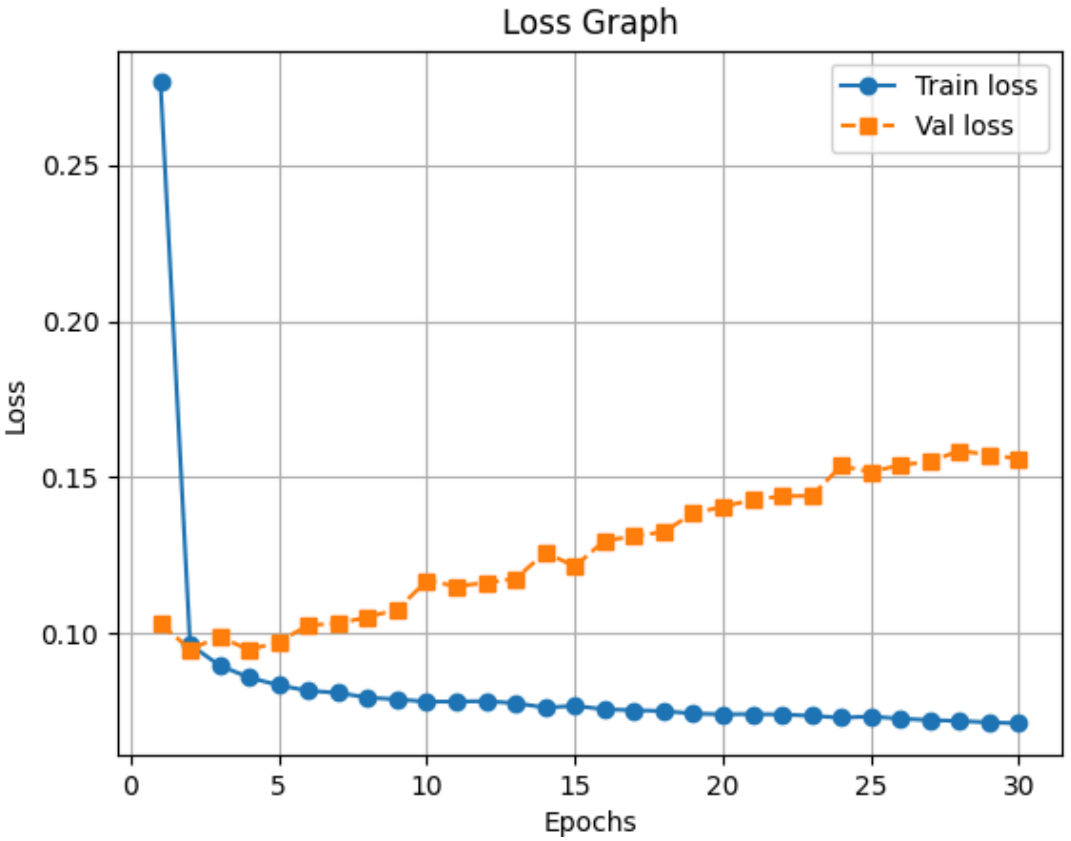
- **Capa de embedding:** transforma las palabras en vectores densos de 100 dimensiones.
- **Capa GRU:** recibe los embeddings como entrada y aprende los patrones contextuales presentes en los textos.
- **Capa densa:** compuesta por 128 neuronas y una capa de salida con 6 neuronas, una por cada emoción.

Para el entrenamiento se utilizó la función de pérdida Cross Entropy, ya que se trata de un problema de clasificación multiclase. La función de activación empleada fue ReLU por su eficiencia computacional, y se estableció una tasa de aprendizaje (learning rate) de 0.001.

El entrenamiento del modelo se llevó a cabo durante 30 épocas. Durante este proceso, se implementó una función que guarda automáticamente el modelo con la menor pérdida en su prueba de validación. Este modelo óptimo se evalúa con test y se generan las gráficas correspondientes a la evolución de la pérdida durante el entrenamiento.

Finalmente, se guardaron los pesos del modelo en un archivo .pt, y tanto el vocabulario como el encoder en archivos .pkl, con el propósito de reutilizar el modelo sin necesidad de volver a entrenarlo.

6. Resultados del modelo

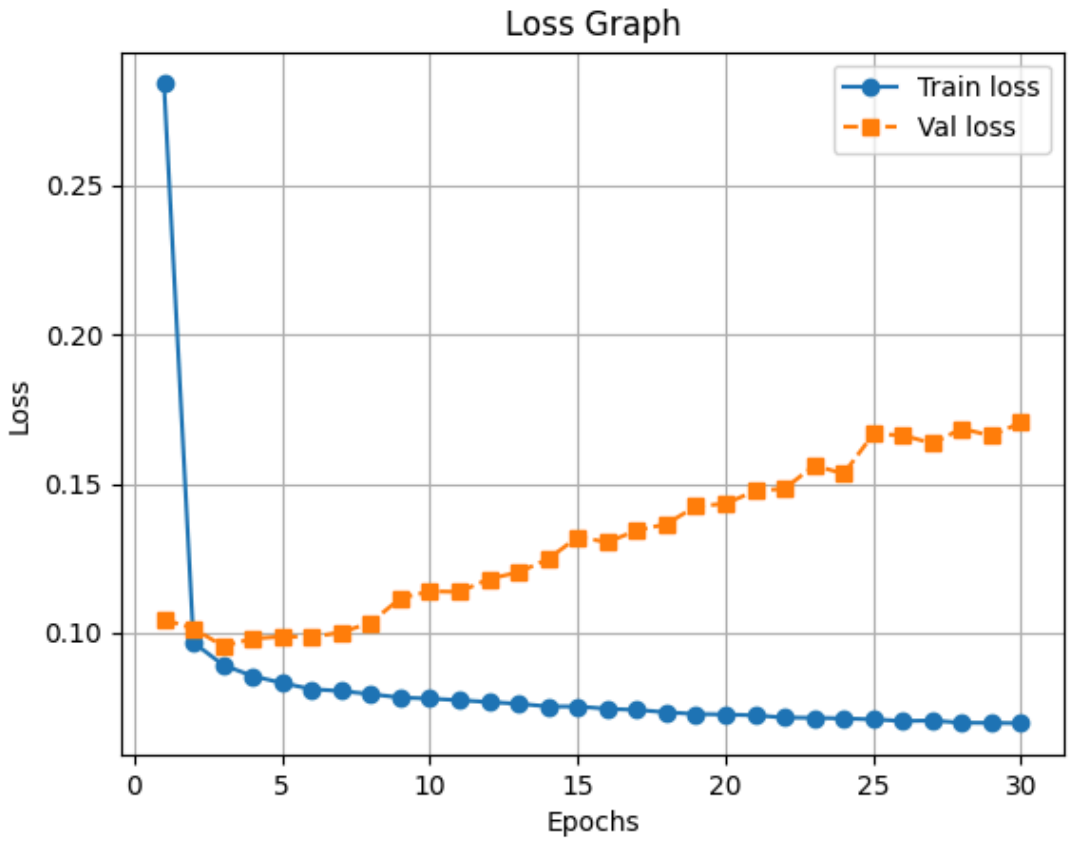


La primera implementación del modelo presentó resultados bastante interesantes. Esto puede observarse en la gráfica de pérdida, donde se aprecia una rápida disminución al inicio del entrenamiento, alcanzando valores cercanos a 0.09. Sin embargo, conforme avanza el proceso de entrenamiento, las curvas de pérdida de entrenamiento y validación comienzan a separarse: mientras la pérdida de entrenamiento continúa disminuyendo, la de validación aumenta, lo que evidencia un claro caso de overfitting. A pesar de ello, el modelo logró un desempeño razonable en su test gracias al uso de una función que conserva el mejor estado del modelo durante el entrenamiento, teniendo los siguientes resultados:

```
=========TEST=========
Pérdida Test: 0.1582 | Precisión Test: 0.9178 | Model Epoch: 4
```

7. Mejoras y resultados finales

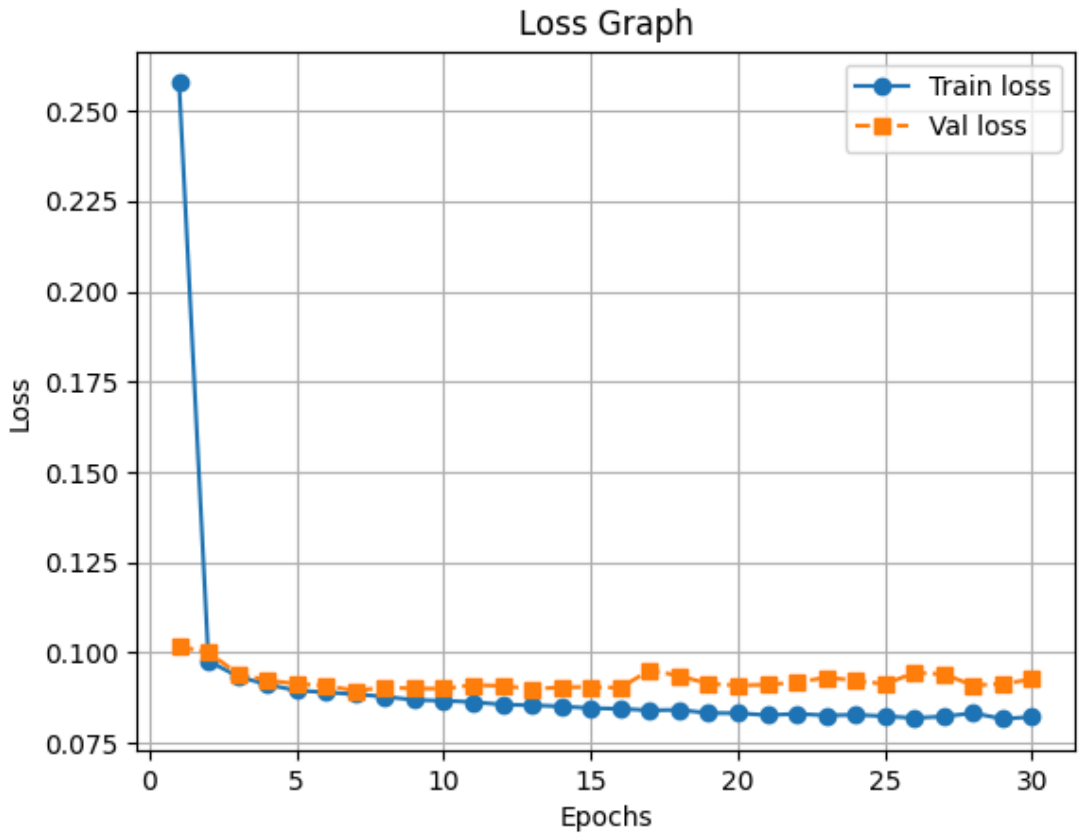
Como se concluyó en el modelo anterior, se presenta un claro caso de overfitting. Para mitigar este problema, se decidió aplicar técnicas de regularización. En primera instancia, se implementó la técnica de dropout con una tasa de 0.5, obteniendo los siguientes resultados.




```
=========TEST=========
Pérdida Test: 0.1757 | Precisión Test: 0.9129 | Model Epoch: 3
```

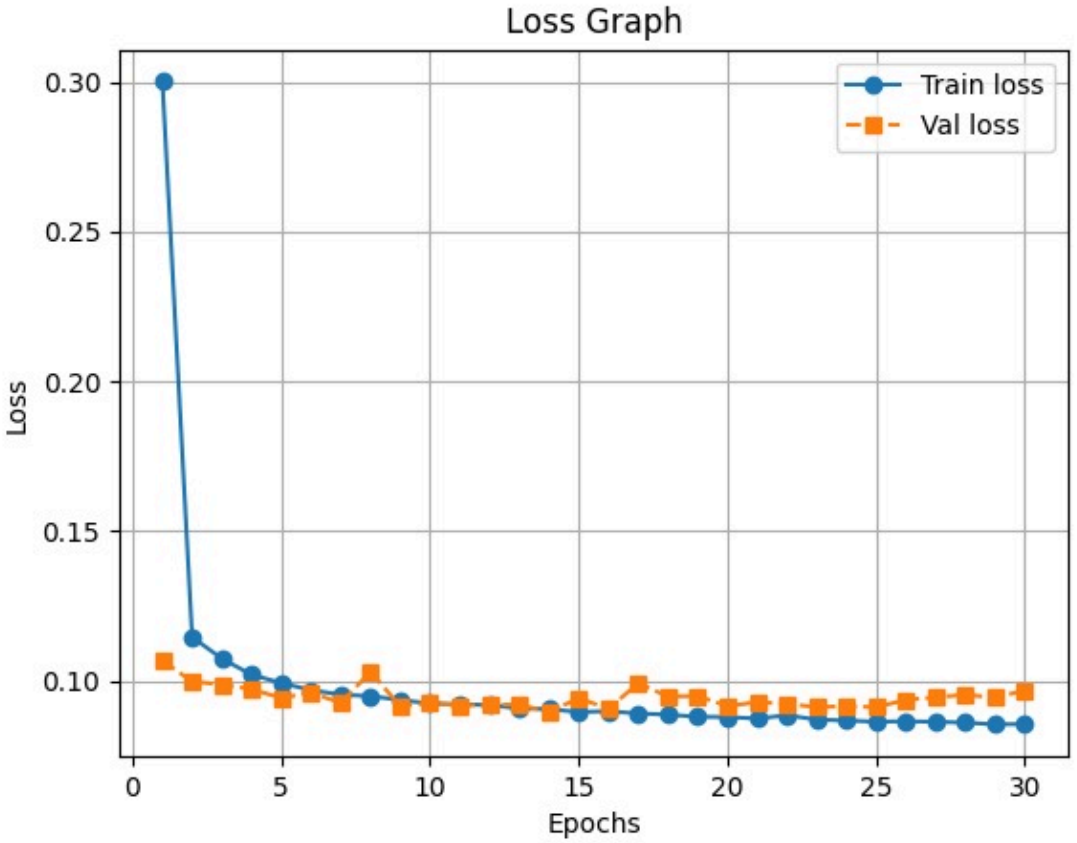
Como podemos observar, el modelo no presenta ninguna mejor, al contrario, se intentó con diferentes valores, pero no ayudo mucho, Dado que el objetivo es reducir lo más posible el problema de overfitting, se decidió cambiar de enfoque e implementar otra técnica de regularización. En este caso, se optó por aplicar Weight Decay con un valor bajo 0.00001.

Esta modificación resultó significativamente más efectiva, ya que produjo los siguientes resultados:




```
=========TEST=========
Pérdida Test: 0.0935 | Precisión Test: 0.9398 | Model Epoch: 7
```

Podemos observar una mejora muy significativa y el modelo ya se puede considerar como fitting, reduciendo la perdida de 0.17 a 0.09, pero pesar de obtener resultados bastante prometedores, se decidió experimentar un poco más e implementar una combinación de ambas técnicas de regularización. Esta mezcla permitió observar un comportamiento más equilibrado del modelo, obteniendo los siguientes resultados:




```
=========TEST=========
Pérdida Test: 0.0966 | Precisión Test: 0.9396 | Model Epoch: 14
```

Se puede observar en la gráfica que la diferencia entre las curvas de validation y train es aún menor, lo que indica una mejora en la generalización del modelo. Sin embargo, los resultados obtenidos en el test siguen mejores superiores en la versión que utiliza únicamente Weight Decay. Por esta razón, se decidió emplear dicha configuración para las pruebas finales de predicción.

8. Predicciones con el modelo

Para realizar las predicciones, se diseñaron frases de prueba correspondientes a cada una de las clases de emociones que se espera que el modelo identifique. Estos fueron los resultados:

```
Usando dispositivo: cuda
Emotions: sadness,joy,love,anger,fear, surprise
Write a sentence with emotion(MAX 32 words):I am very happy today
Emoción Predicha: joy
Write a sentence with emotion(MAX 32 words):I feel lonely and tired.
Emoción Predicha: sadness
Write a sentence with emotion(MAX 32 words):I am extremely frustrated with this situation.
Emoción Predicha: anger
Write a sentence with emotion(MAX 32 words):I feel scared to go outside at night.
Emoción Predicha: fear
Write a sentence with emotion(MAX 32 words):I did not expect that to happen.
Emoción Predicha: surprise
Write a sentence with emotion(MAX 32 words):I feel deeply in love with you.
Emoción Predicha: love
Write a sentence with emotion(MAX 32 words):I love this movie
Emoción Predicha: fear
```

Los resultados obtenidos fueron bastante satisfactorios, ya que el modelo clasificó correctamente 6 de las 7 frases propuestas. Sin embargo, aún no se ha determinado la causa del error en la última predicción. A pesar de haber reentrenado el modelo en múltiples ocasiones e incorporado ejemplos adicionales en el conjunto de datos con situaciones similares a dicha frase, el modelo continúa clasificándola como miedo, aunque alegría o amor serían categorías más adecuadas.

9. Conclusión

El desarrollo del modelo permitió comprender y aplicar técnicas de procesamiento de lenguaje natural orientadas al reconocimiento de emociones en texto. A lo largo del proceso, se implementaron distintas estrategias de regularización, como Dropout y Weight Decay, con el objetivo de mejorar la capacidad de generalización del modelo y mitigar el overfitting. Los

resultados obtenidos, a pesar de demostrar algunas limitaciones y áreas de mejora, muestran un desempeño sólido, logrando clasificar correctamente la mayoría de las frases de prueba, con una precisión notable en la mayoría de las emociones planteadas.