Red Neuronal para la Predicción de Partidos de Fútbol Internacional

Red neuronal diseñada para predecir el resultado de partidos de fútbol internacionales, con el objetivo de simular torneos como la Copa del Mundo 2026. El modelo utiliza una arquitectura avanzada con capas de Embedding para aprender las características de cada selección nacional a partir de un extenso dataset histórico de más de 40,000 partidos. Al recibir los nombres de dos equipos, el modelo calcula las probabilidades de los tres resultados posibles: victoria del equipo 1, empate o victoria del equipo 2.

2. Objetivo del Proyecto

El objetivo principal es crear un modelo predictivo capaz de determinar el resultado más probable de un partido de fútbol entre dos selecciones nacionales. El sistema debe ser capaz de:

- Aprender patrones de rendimiento a partir de datos históricos.
- Cuantificar la "fuerza" o "perfil" de cada equipo de manera implícita.
- Proporcionar una salida probabilística para simular torneos completos.

3. Descripción de los Datos

- Fuente: Se utilizó un dataset público de Kaggle que contiene resultados de partidos internacionales desde 1872. Para este modelo, se filtraron los datos para incluir únicamente partidos disputados a partir de 1990, con el fin de reflejar el fútbol moderno.
- Características de Entrada (input): Los identificadores numéricos (IDs) del equipo local y del equipo visitante.
- Variable Objetivo (output): El resultado final del partido, codificado en tres categorías:
    - 0: Victoria del equipo visitante.
    - 1: Empate.
    - 2: Victoria del equipo local.

4. Arquitectura del Modelo

Se implementó una red neuronal utilizando la API Funcional de Keras, lo que permite una mayor flexibilidad al definir arquitecturas complejas con múltiples entradas.

1. Entradas Múltiples: El modelo tiene dos entradas separadas: una para el ID del equipo local y otra para el ID del equipo visitante.
2. Capa de Embedding: Esta es la capa clave del modelo.
    - Función: Transforma el ID numérico de cada equipo (ej: Brasil = 25) en un vector denso de números (ej: [0.7, -0.1, 0.4, ...]). Este vector, llamado

"embedding", es un perfil numérico que representa la fuerza, el estilo y el rendimiento histórico del equipo. El modelo aprende estos perfiles automáticamente durante el entrenamiento.

- Ventaja: En lugar de tratar a cada equipo como una simple categoría, el modelo puede entender relaciones complejas (ej: los vectores de Brasil y Argentina podrían ser numéricamente más similares entre sí que los de Brasil y San Marino).

3. Concatenación: Los dos vectores de embedding (uno por equipo) se unen para formar un único vector que representa el enfrentamiento del partido.
4. Capas Densas: Este vector combinado pasa a través de capas densas (Dense) con activación ReLU y capas de regularización (Dropout) para analizar la interacción entre los perfiles de los equipos y prevenir el sobreajuste.
5. Capa de Salida: Una capa final Dense con 3 neuronas y una función de activación Softmax. Esta capa produce el resultado final: un vector de tres valores que representan la probabilidad de cada resultado, y cuya suma siempre es 1.

5. Implementación y Uso

El modelo se implementó en Python utilizando las librerías TensorFlow/Keras y Scikit-learn.

Función de Predicción

La siguiente función encapsula la lógica para predecir un partido:

```
Tabnine | Edit | Test | Explain | Document
def predict_match(team1, team2, model, encoder):
    """
    Predice el resultado de un partido entre dos equipos.
    Devuelve las probabilidades: [Victoria Team2, Empate, Victoria Team1]
    """
    team1_encoded = encoder.transform([team1])
    team2_encoded = encoder.transform([team2])

    prediction = model.predict([team1_encoded, team2_encoded], verbose=0)
    return prediction[0]

team1 = 'Spain'
team2 = 'Morocco'
probabilities = predict_match(team1, team2, model, team_encoder)

print(f"\n--- Predicción para {team1} vs. {team2} ---")
print(f"Prob. Victoria {team1}: {probabilities[2] * 100:.2f}%")
print(f"Prob. Empate: {probabilities[1] * 100:.2f}%")
print(f"Prob. Victoria {team2}: {probabilities[0] * 100:.2f}%")
✓ 0.1s
```

6. Limitaciones y Próximos Pasos

- Simplicidad: El modelo actual no considera factores importantes como la forma reciente de los equipos, jugadores lesionados, la importancia del torneo o la ubicación del partido (más allá de local/visitante).
- Datos Históricos: El rendimiento pasado no siempre es un indicador perfecto del rendimiento futuro.

Para mejorar el modelo, se podrían incorporar características adicionales como el ranking FIFA de los equipos, resultados de los últimos 5 partidos o estadísticas de goles a favor y en contra.

Referencias usadas

Capas de Embedding (Keras)

- Referencia: Guía de Keras sobre clasificación con datos estructurados
- Concepto: Es la técnica principal para convertir una variable categórica (como el nombre de un equipo) en un vector numérico que el modelo puede entender.

Arquitectura de Múltiples Entradas (Keras)

- Referencia: Guía de la API Funcional de Keras
- Concepto: Permite construir modelos con arquitecturas complejas, como la que usamos con una entrada para el equipo local y otra para el visitante.

Dataset de Partidos Internacionales

- Referencia: International football results from 1872 to 2023 en Kaggle
- Concepto: Es la fuente de datos históricos utilizada para entrenar el modelo.

Tutoriales de la Comunidad

- Búsqueda recomendada: "Predicting sports with Keras embedding" o "match outcome prediction neural network".
- Concepto: Proyectos y artículos de otros desarrolladores que aplican técnicas similares a problemas de predicción deportiva.