COLOMBIA POTENCIA DE LA VIDA

TIC

Actividad 2

Modelos de redes neuronales en Keras

TALENTO TECH

AZ | PROYECTOS EDUCATIVOS

UTP Universidad Tecnológica de Pereira

Modelos de redes neuronales en Keras

La documentacion oficial de keras la pueden encontrar en: https://keras.io/api/

Biblioteca Keras

Keras es una de las bibliotecas de aprendizaje profundo más populares y ampliamente utilizadas en el campo de las redes neuronales. Es una biblioteca de código abierto escrita en Python que proporciona una interfaz amigable y de alto nivel para la construcción, entrenamiento y evaluación de modelos de redes neuronales de manera rápida y eficiente.

La importancia de Keras en el campo de las redes neuronales radica en varios aspectos:

Facilidad de uso

Keras ofrece una interfaz de programación simple y coherente que permite a los usuarios construir y experimentar con modelos de redes neuronales de forma intuitiva. Está diseñado para ser accesible tanto para principiantes como para expertos, lo que facilita la entrada al campo del aprendizaje profundo.

Modelos de redes neuronales en Keras

La importancia de Keras en el campo de las redes neuronales radica en varios aspectos:

Flexibilidad:
Keras es altamente flexible y permite la construcción de una amplia variedad de arquitecturas de redes neuronales, desde modelos simples hasta redes profundas y complejas. Permite la creación de modelos secuenciales (capa por capa) o modelos más complejos con múltiples entradas y salidas.

Compatibilidad con otras bibliotecas:
Keras se integra perfectamente con otras bibliotecas populares de aprendizaje profundo, como TensorFlow y Theano. Esto significa que los usuarios pueden aprovechar toda la potencia de estas bibliotecas subyacentes mientras disfrutan de la simplicidad y la facilidad de uso de Keras.

Modelos de redes neuronales en Keras

La importancia de Keras en el campo de las redes neuronales radica en varios aspectos:

Gran comunidad y soporte:
Keras cuenta con una gran comunidad de usuarios y desarrolladores que contribuyen activamente con ejemplos, tutoriales, documentación y código abierto. Esto proporciona un valioso recurso para aquellos que están aprendiendo y trabajando en el campo del aprendizaje profundo.

Keras es una biblioteca esencial en el campo de las redes neuronales debido a su facilidad de uso, flexibilidad, compatibilidad con otras bibliotecas, gran comunidad y adopción industrial. Facilita la experimentación y el desarrollo de modelos de aprendizaje profundo, permitiendo a los usuarios centrarse en la construcción de soluciones innovadoras en lugar de preocuparse por los detalles de implementación.

El proceso para implementar una red neuronal utilizando Keras sigue varios pasos generales

1. Preparación de los datos:

Asegúrate de tener tus datos correctamente estructurados en matrices NumPy u otros formatos compatibles con Keras. Divide tus datos en conjuntos de entrenamiento, validación y prueba si es necesario.

2. Importación de la biblioteca:

Importa las clases y funciones necesarias de Keras, incluyendo Sequential para modelos secuenciales y capas específicas como Dense, Conv2D, LSTM, etc., dependiendo del tipo de red neuronal que estés construyendo.

El proceso para implementar una red neuronal utilizando Keras sigue varios pasos generales

3. Configuración del modelo

Crea una instancia de un modelo secuencial (Sequential) o funcional de Keras.
Añade capas al modelo, especificando el número de neuronas, la función de activación, la regularización, entre otros, según tus necesidades. Puedes hacerlo usando el método add del modelo.

4. Compilación del modelo

Utiliza el método compile para compilar el modelo, especificando el optimizador, la función de pérdida y las métricas que se utilizarán para evaluar el rendimiento del modelo durante el entrenamiento.

El proceso para implementar una red neuronal utilizando Keras sigue varios pasos generales

5. Entrenamiento del modelo

Utiliza el método fit para entrenar el modelo con los datos de entrenamiento. Especifica el número de épocas de entrenamiento y el tamaño del lote (batch size).

6. Evaluación del modelo

Utiliza el método evaluate para evaluar el rendimiento del modelo en los datos de validación o prueba.

7. Predicciones

Utiliza el método predict para realizar predicciones en nuevos datos utilizando el modelo entrenado.

El proceso para implementar una red neuronal utilizando Keras sigue varios pasos generales

8. Ajuste de Hiperparámetros

Opcionalmente, puedes realizar un ajuste de hiperparámetros utilizando técnicas como la búsqueda de cuadrícula (GridSearchCV) o la búsqueda aleatoria (RandomizedSearchCV) para encontrar la combinación óptima de hiperparámetros que maximice el rendimiento del modelo.

Es importante tener en cuenta que la implementación exacta puede variar según el tipo de red neuronal que estés construyendo (redes totalmente conectadas, convolucionales, recurrentes, etc.) y los requisitos específicos de tu problema. Sin embargo, estos pasos generales proporcionan una guía básica para implementar redes neuronales utilizando Keras.

Ejemplo básico de cómo implementar una red neuronal utilizando Keras en Python

En este ejemplo:

- **Importación de bibliotecas:**

```
import numpy as np
import keras
```

Importamos las bibliotecas necesarias. keras es una API de alto nivel para construir y entrenar modelos de redes neuronales.

Ejemplo básico de cómo implementar una red neuronal utilizando Keras en Python

- **Construcción del modelo:**

```
# Build a simple Sequential model+

model = keras.Sequential([keras.layers.Dense(units=1,
input_shape=[1])])
```

Creamos un modelo secuencial utilizando Sequential de Keras, lo que significa que las capas se apilan en secuencia.
Agregamos una capa densa (Dense) al modelo. Esta capa tiene una sola unidad (units=1) y espera una entrada de una dimensión (input_shape=[1]).

Ejemplo básico de cómo implementar una red neuronal utilizando Keras en Python

- **Compilación del modelo:**

```
# Compile the model
model.compile(optimizer='sgd', loss='mean_squared_error')
```

Compilamos el modelo utilizando el optimizador de descenso de gradiente estocástico (sgd) y la función de pérdida de error cuadrático medio (mean_squared_error). Esto configura el modelo para la fase de entrenamiento.

Ejemplo básico de cómo implementar una red neuronal utilizando Keras en Python

- **Declaración de entradas y salidas para el entrenamiento:**

Para esta ejemplo vamos a asumir que tenemos como entradas X los números del 1 al 5 y como salidas Y vamos asumir que tenemos una función que toma cada x la multiplica por 10 y le suma 1 asi:

x	y
1	11
2	21
3	31
4	41
5	51

```
# Declare model inputs and outputs for training
X = np.array([1.0, 2.0, 3.0, 4.0, 5.0], dtype=float)
y = np.array([11.0,21.0,31.0,41.0,51.0], dtype=float)
```

Declaramos los datos de entrada (X) y los resultados deseados (y) para el entrenamiento del modelo.

Ejemplo básico de cómo implementar una red neuronal utilizando Keras en Python

- **Entrenamiento del modelo:**

```
# Train the model
model.fit(X, y, epochs=100)
```

Entrenamos el modelo utilizando los datos de entrada (X) y los resultados deseados (y) durante 100 épocas. Durante cada época, el modelo ajusta sus pesos para minimizar la función de pérdida especificada en el paso de compilación.

Ejemplo básico de cómo implementar una red neuronal utilizando Keras en Python

- **Realización de una predicción:**

```
# Make a prediction
print(model.predict([10.0]))
```

Realizamos una predicción utilizando el modelo entrenado, pasando una entrada de prueba de 10.0. El modelo utiliza los pesos ajustados durante el entrenamiento para predecir el resultado correspondiente.

Este código implementa y entrena un modelo de red neuronal muy simple utilizando Keras. El modelo tiene una sola capa densa y se entrena para predecir una salida a partir de una única entrada.

COLOMBIA POTENCIA DE LA VIDA

TIC

TALENTO TECH

AZ | PROYECTOS EDUCATIVOS

UTP Universidad Tecnológica de Pereira