```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
import numpy as np
```

```
text_data = """
Artificial intelligence is the simulation of human intelligence in machines.
AI systems can perform tasks such as learning reasoning and problem solving.
Machine learning is a subset of artificial intelligence.
Deep learning uses neural networks with multiple layers.
Artificial intelligence is widely used in healthcare.
AI improves decision making in business.

Cyber security protects systems and data from cyber attacks.
Hackers use malware phishing and ransomware.
Firewalls and antivirus software provide protection.
Strong passwords prevent unauthorized access.

Data science involves analyzing large amounts of data.
Python is widely used in data science.
Data visualization helps understand data.
"""
```

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts([text_data])

total_words = len(tokenizer.word_index) + 1
```

```
input_sequences = []

for line in text_data.split("\n"):
    token_list = tokenizer.texts_to_sequences([line])[0]
    for i in range(1, len(token_list)):
        input_sequences.append(token_list[:i+1])
```

```
max_sequence_len = max(len(seq) for seq in input_sequences)

input_sequences = pad_sequences(
    input_sequences,
    maxlen=max_sequence_len,
    padding='pre'
)
```

```
X = input_sequences[:, :-1]
y = input_sequences[:, -1]

y = tf.keras.utils.to_categorical(y, num_classes=total_words)
```

```
model = Sequential()
model.add(Embedding(total_words, 64, input_length=max_sequence_len - 1))
model.add(LSTM(100))
model.add(Dense(total_words, activation='softmax'))

model.compile(
    loss='categorical_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)

model.summary()
```

```
/usr/local/lib/python3.12/dist-packages/keras/src/layers/core/embedding.py:97: UserWarning: Argument `input_length` is depre
  warnings.warn(
```

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	?	0 (unbuilt)
lstm (LSTM)	?	0 (unbuilt)
dense (Dense)	?	0 (unbuilt)

Total params: 0 (0.00 B)
Trainable params: 0 (0.00 B)
Non-trainable params: 0 (0.00 B)

```
model.fit(X, y, epochs=200, verbose=1)
```

```
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9665 - loss: 0.2275
Epoch 173/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9449 - loss: 0.2409
Epoch 174/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.9626 - loss: 0.2428
Epoch 175/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9606 - loss: 0.2141
Epoch 176/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.9488 - loss: 0.2218
Epoch 177/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step - accuracy: 0.9626 - loss: 0.2220
Epoch 178/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9410 - loss: 0.2309
Epoch 179/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9488 - loss: 0.2166
Epoch 180/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.9743 - loss: 0.2003
Epoch 181/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step - accuracy: 0.9606 - loss: 0.1923
Epoch 182/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.9704 - loss: 0.2063
Epoch 183/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.9410 - loss: 0.2156
Epoch 184/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9782 - loss: 0.1816
Epoch 185/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9743 - loss: 0.1808
Epoch 186/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.9665 - loss: 0.2022
Epoch 187/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9470 - loss: 0.2025
Epoch 188/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step - accuracy: 0.9470 - loss: 0.2123
Epoch 189/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9782 - loss: 0.1798
Epoch 190/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9548 - loss: 0.1858
Epoch 191/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step - accuracy: 0.9567 - loss: 0.1755
Epoch 192/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9449 - loss: 0.1863
Epoch 193/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9312 - loss: 0.1932
Epoch 194/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step - accuracy: 0.9371 - loss: 0.1997
Epoch 195/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9606 - loss: 0.1696
Epoch 196/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9665 - loss: 0.1737
Epoch 197/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9470 - loss: 0.1907
Epoch 198/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9548 - loss: 0.1753
Epoch 199/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.9743 - loss: 0.1578
Epoch 200/200
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step - accuracy: 0.9704 - loss: 0.1628
<keras.src.callbacks.history.History at 0x7d9067f8b260>
```

```
def generate_text(seed_text, next_words=40):
    for _ in range(next_words):
        token_list = tokenizer.texts_to_sequences([seed_text])[0]
        token_list = pad_sequences(
            [token_list],
            maxlen=max_sequence_len - 1,
            padding='pre'
        )

        predicted = np.argmax(
            model.predict(token_list, verbose=0),
```

```
            axis=-1
        )

        output_word = ""
        for word, index in tokenizer.word_index.items():
            if index == predicted:
                output_word = word
                break

        seed_text += " " + output_word

    return seed_text
```

```
print(generate_text("artificial intelligence"))
```

```
ligence is widely used in healthcare healthcare machines machines machines machines solving solving solving solving solving s
```

```
print(generate_text("cyber security"))
```

```
cyber security protects systems and data from cyber attacks attacks solving solving solving solving solving solving solving
```

```
print(generate_text("data science"))
```

```
data science involves analyzing large amounts of data data attacks attacks attacks attacks attacks machines machines solving
```