Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

Recurrent neural networks

Practice Session 12

Changho Suh

January 25, 2024

Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

Outline

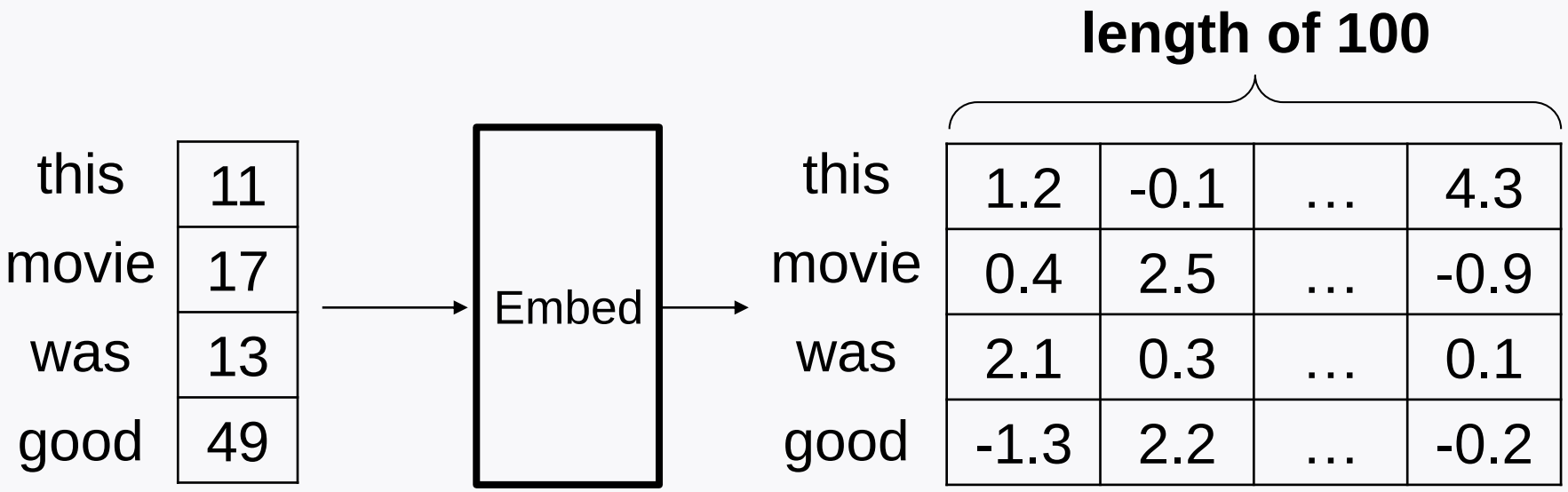
Will implement two RNN models:

1. Basic RNN

2. LSTM

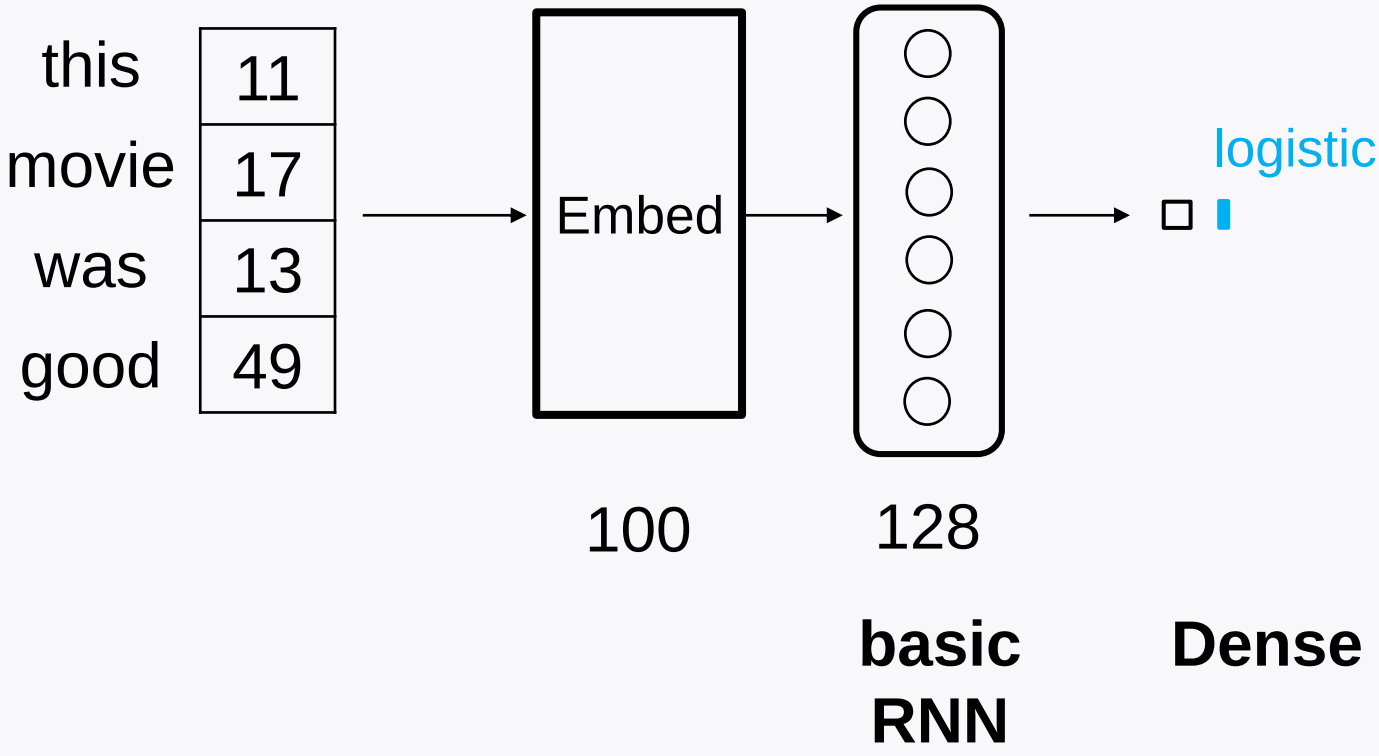
Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

Basic RNN



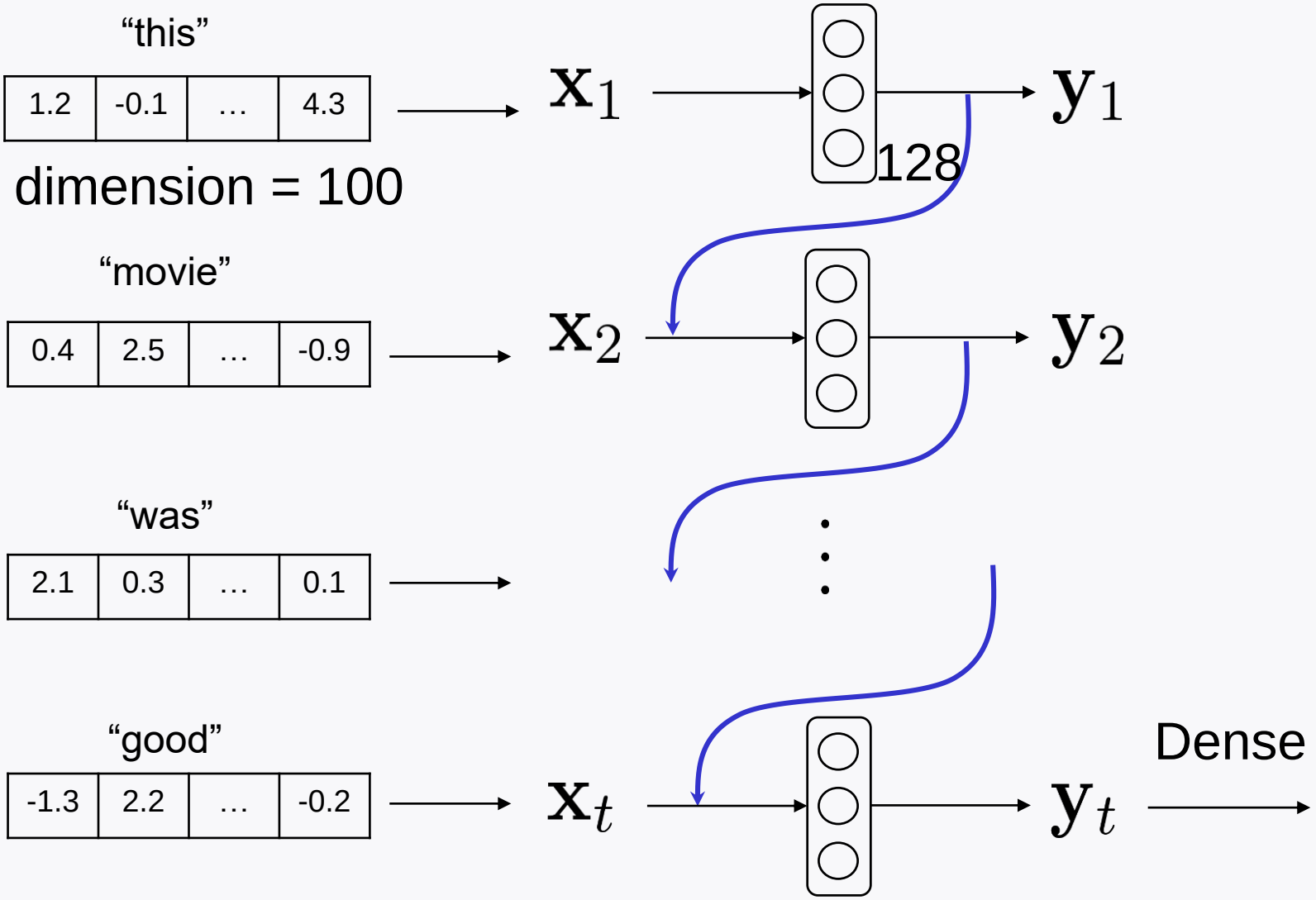
Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

Basic RNN



Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

Basic RNN: Unrolled version



Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

Loading IMDB

```
from tensorflow.keras.datasets import imdb

(X_train,y_train), (X_test,y_test) = imdb.load_data(num_words=10000)

from tensorflow.keras.preprocessing.sequence import pad_sequences

max_len = 256
X_train_pad = pad_sequences(X_train,value=0, padding='post', maxlen=max_len)
X_test_pad = pad_sequences(X_test, value=0, padding='post',maxlen=max_len)
```

Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

Basic RNN: Tensorflow coding

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Embedding
from tensorflow.keras.layers import SimpleRNN

basic_rnn = Sequential()
basic_rnn.add(Embedding(input_dim=10000, output_dim=100, input_length=256))
basic_rnn.add(SimpleRNN(128))
basic_rnn.add(Dense(1,activation='sigmoid'))

basic_rnn.compile(optimizer='adam',loss='binary_crossentropy',metrics='acc')

history_basic_rnn = basic_rnn.fit(X_train_pad,y_train, epochs=10)
```

Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

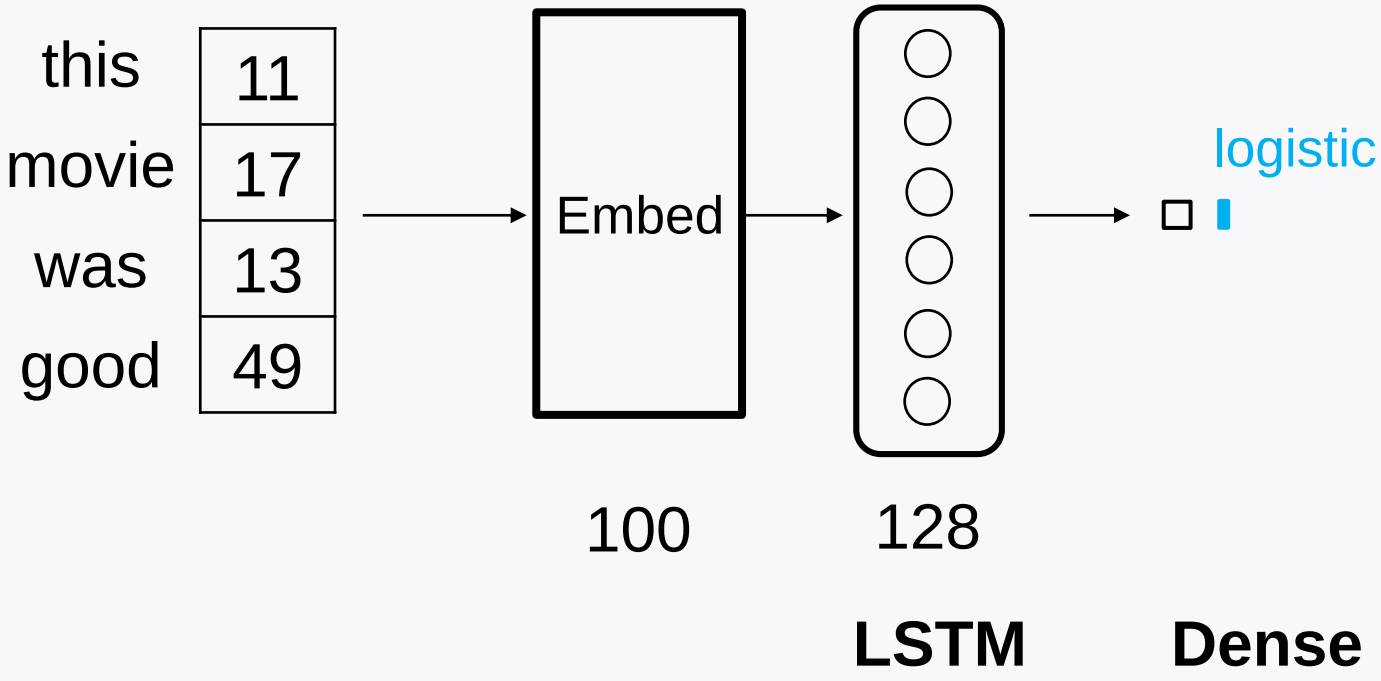
Basic RNN: Performance

```
test_performance = basic_rnn.evaluate(X_test_pad,y_test)
print(test_performance)
```

```
782/782 [==============================] - 8s 10ms/step - loss: 0.7149 - acc:
0.5500
[0.7149398922920227, 0.5500400066375732]
```

Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

LSTM



Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

LSTM: Tensorflow coding

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Embedding
from tensorflow.keras.layers import LSTM

LSTM_model = Sequential()
LSTM_model.add(Embedding(input_dim=10000,output_dim=100,input_length=256))
LSTM_model.add(LSTM(128))
LSTM_model.add(Dense(1,activation='sigmoid'))

LSTM_model.compile(optimizer='adam',
                   loss='binary_crossentropy',
                   metrics='acc')

LSTM_model.fit(X_train_pad,y_train, epochs=10)
```

Copyright © 2024 Changho Suh @ KAIST. All rights reserved.

LSTM: Performance

```
test_performance = LSTM_model.evaluate(X_test_pad,y_test)
print(test_performance)
```

```
loss: 0.4696 - acc: 0.8592
```