Homework 2

May 14, 2019

```
In [1]: import pandas as pd
        from keras.preprocessing.text import Tokenizer
        from keras.preprocessing.sequence import pad_sequences
        from keras.utils import to_categorical
```

```
Using TensorFlow backend.
```

1. *Import the data and tokenize to use with Keras.*

```
In [35]: train_df = pd.read_csv('data/congress_train.csv', encoding='ISO-8859-1')
         valid_df = pd.read_csv('data/congress_val.csv', encoding='ISO-8859-1')
         test_df = pd.read_csv('data/congress_test.csv', encoding='ISO-8859-1')
```

```
In [50]: train_words = []
         valid_words = []
         test_words = []
         for title in list(train_df['Title']):
             train_words.append(str(title))
         for title in list(valid_df['Title']):
             valid_words.append(str(title))
         for title in list(valid_df['Title']):
             test_words.append(str(title))
```

```
In [51]: train_y = to_categorical(list(train_df['Major']))
         valid_y = to_categorical(list(valid_df['Major']))
         test_y = to_categorical(list(valid_df['Major']))
```

```
In [52]: #Keep only the 10000 most frequent words
         tokenizer = Tokenizer(num_words=10000)
         tokenizer.fit_on_texts(train_words)
```

```
In [53]: #Limit each bill's title to a maximum length of 100 words
         #Pad each sequence to be of length 100
         train_seq = tokenizer.texts_to_sequences(train_words)
         test_seq = tokenizer.texts_to_sequences(test_words)
         valid_seq = tokenizer.texts_to_sequences(valid_words)
```

```
In [54]: train_x = pad_sequences(train_seq, maxlen=100)
         test_x = pad_sequences(test_seq, maxlen=100)
         valid_x = pad_sequences(valid_seq, maxlen=100)
```

2. *Use a task-specific embedding layer with an appropriate number of output dimensions*
3. *Estimate a basic feed-forward network*

```
In [40]: from keras.layers import Embedding, Flatten, Dense
         from keras.models import Sequential
```

```
In [81]: ffn = Sequential()
         ffn.add(Embedding(10000, 25, input_length=100))
         ffn.add(Flatten())
         ffn.add(Dense(24, activation='softmax'))
         ffn.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']
         result_ffn = ffn.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs=50, b
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/50
278612/278612 [==============================] - 7s 27us/step - loss: 1.8746 - acc: 0.4891 - va
Epoch 2/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.8514 - acc: 0.7779 - va
Epoch 3/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.6629 - acc: 0.8213 - va
Epoch 4/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.5896 - acc: 0.8399 - va
Epoch 5/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.5447 - acc: 0.8509 - va
Epoch 6/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.5128 - acc: 0.8585 - va
Epoch 7/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.4879 - acc: 0.8657 - va
Epoch 8/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.4673 - acc: 0.8709 - va
Epoch 9/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.4496 - acc: 0.8757 - va
Epoch 10/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.4343 - acc: 0.8801 - va
Epoch 11/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.4207 - acc: 0.8838 - va
Epoch 12/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.4085 - acc: 0.8868 - va
Epoch 13/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.3975 - acc: 0.8896 - va
Epoch 14/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.3871 - acc: 0.8924 - va
Epoch 15/50
278612/278612 [==============================] - 7s 24us/step - loss: 0.3779 - acc: 0.8949 - va
Epoch 16/50
```

```
278612/278612 [==============================] - 6s 23us/step - loss: 0.3693 - acc: 0.8974 - va
Epoch 17/50
278612/278612 [==============================] - 6s 23us/step - loss: 0.3613 - acc: 0.8993 - va
Epoch 18/50
278612/278612 [==============================] - 6s 23us/step - loss: 0.3539 - acc: 0.9012 - va
Epoch 19/50
278612/278612 [==============================] - 6s 23us/step - loss: 0.3468 - acc: 0.9033 - va
Epoch 20/50
278612/278612 [==============================] - 6s 21us/step - loss: 0.3404 - acc: 0.9047 - va
Epoch 21/50
278612/278612 [==============================] - 5s 20us/step - loss: 0.3342 - acc: 0.9060 - va
Epoch 22/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.3285 - acc: 0.9078 - va
Epoch 23/50
278612/278612 [==============================] - 5s 20us/step - loss: 0.3231 - acc: 0.9091 - va
Epoch 24/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.3180 - acc: 0.9106 - va
Epoch 25/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.3132 - acc: 0.9117 - va
Epoch 26/50
278612/278612 [==============================] - 6s 21us/step - loss: 0.3086 - acc: 0.9130 - va
Epoch 27/50
278612/278612 [==============================] - 5s 20us/step - loss: 0.3045 - acc: 0.9142 - va
Epoch 28/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.3004 - acc: 0.9150 - va
Epoch 29/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.2966 - acc: 0.9159 - va
Epoch 30/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.2929 - acc: 0.9170 - va
Epoch 31/50
278612/278612 [==============================] - 6s 20us/step - loss: 0.2894 - acc: 0.9181 - va
Epoch 32/50
278612/278612 [==============================] - 6s 23us/step - loss: 0.2861 - acc: 0.9188 - va
Epoch 33/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2829 - acc: 0.9197 - va
Epoch 34/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2799 - acc: 0.9206 - va
Epoch 35/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2771 - acc: 0.9214 - va
Epoch 36/50
278612/278612 [==============================] - 6s 21us/step - loss: 0.2744 - acc: 0.9218 - va
Epoch 37/50
278612/278612 [==============================] - 6s 21us/step - loss: 0.2717 - acc: 0.9229 - va
Epoch 38/50
278612/278612 [==============================] - 6s 21us/step - loss: 0.2692 - acc: 0.9233 - va
Epoch 39/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2668 - acc: 0.9240 - va
Epoch 40/50
```

```
278612/278612 [==============================] - 6s 21us/step - loss: 0.2644 - acc: 0.9248 - va
Epoch 41/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2623 - acc: 0.9252 - va
Epoch 42/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2601 - acc: 0.9261 - va
Epoch 43/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2581 - acc: 0.9263 - va
Epoch 44/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2560 - acc: 0.9270 - va
Epoch 45/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2542 - acc: 0.9275 - va
Epoch 46/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2523 - acc: 0.9279 - va
Epoch 47/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2505 - acc: 0.9283 - va
Epoch 48/50
278612/278612 [==============================] - 6s 22us/step - loss: 0.2488 - acc: 0.9290 - va
Epoch 49/50
278612/278612 [==============================] - 7s 24us/step - loss: 0.2472 - acc: 0.9293 - va
Epoch 50/50
278612/278612 [==============================] - 7s 24us/step - loss: 0.2455 - acc: 0.9296 - va
```

4. *Estimate a recurrent neural network (RNN) with a layer_simple_rnn*

In [57]:
```
from keras.layers import SimpleRNN
```

In [59]:
```
rnn = Sequential()
rnn.add(Embedding(10000, 25, input_length=100))
rnn.add(SimpleRNN(25))
rnn.add(Dense(24, activation='softmax'))
rnn.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']
result_rnn = rnn.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs=50, b
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/50
278612/278612 [==============================] - 27s 96us/step - loss: 2.3740 - acc: 0.3132 - v
Epoch 2/50
278612/278612 [==============================] - 26s 95us/step - loss: 1.4785 - acc: 0.5869 - v
Epoch 3/50
278612/278612 [==============================] - 27s 95us/step - loss: 1.2144 - acc: 0.6779 - v
Epoch 4/50
278612/278612 [==============================] - 26s 95us/step - loss: 1.0309 - acc: 0.7342 - v
Epoch 5/50
278612/278612 [==============================] - 26s 95us/step - loss: 0.9261 - acc: 0.7626 - v
Epoch 6/50
278612/278612 [==============================] - 27s 95us/step - loss: 0.8646 - acc: 0.7785 - v
Epoch 7/50
278612/278612 [==============================] - 26s 95us/step - loss: 0.8078 - acc: 0.7944 - v
```

```
Epoch 8/50
278612/278612 [==============================] - 27s 95us/step - loss: 0.7778 - acc: 0.8017 - v
Epoch 9/50
278612/278612 [==============================] - 29s 104us/step - loss: 0.7408 - acc: 0.8104 -
Epoch 10/50
278612/278612 [==============================] - 28s 99us/step - loss: 0.7161 - acc: 0.8168 - v
Epoch 11/50
278612/278612 [==============================] - 26s 95us/step - loss: 0.6918 - acc: 0.8228 - v
Epoch 12/50
278612/278612 [==============================] - 26s 95us/step - loss: 0.6746 - acc: 0.8271 - v
Epoch 13/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.6544 - acc: 0.8319 - v
Epoch 14/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.6330 - acc: 0.8364 - v
Epoch 15/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.6154 - acc: 0.8417 - v
Epoch 16/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.5991 - acc: 0.8457 - v
Epoch 17/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.5981 - acc: 0.8460 - v
Epoch 18/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.5752 - acc: 0.8517 - v
Epoch 19/50
278612/278612 [==============================] - 27s 97us/step - loss: 0.5644 - acc: 0.8536 - v
Epoch 20/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.5516 - acc: 0.8575 - v
Epoch 21/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.5403 - acc: 0.8602 - v
Epoch 22/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.5291 - acc: 0.8631 - v
Epoch 23/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.5224 - acc: 0.8646 - v
Epoch 24/50
278612/278612 [==============================] - 27s 97us/step - loss: 0.5168 - acc: 0.8665 - v
Epoch 25/50
278612/278612 [==============================] - 26s 93us/step - loss: 0.5070 - acc: 0.8690 - v
Epoch 26/50
278612/278612 [==============================] - 246s 882us/step - loss: 0.4978 - acc: 0.8710 -
Epoch 27/50
278612/278612 [==============================] - 421s 2ms/step - loss: 0.5011 - acc: 0.8700 - v
Epoch 28/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.4844 - acc: 0.8745 - v
Epoch 29/50
278612/278612 [==============================] - 27s 95us/step - loss: 0.4787 - acc: 0.8757 - v
Epoch 30/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.4724 - acc: 0.8780 - v
Epoch 31/50
278612/278612 [==============================] - 27s 97us/step - loss: 0.4713 - acc: 0.8783 - v
```

```
Epoch 32/50
278612/278612 [==============================] - 27s 97us/step - loss: 0.4683 - acc: 0.8784 - v
Epoch 33/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.4596 - acc: 0.8805 - v
Epoch 34/50
278612/278612 [==============================] - 27s 96us/step - loss: 0.4531 - acc: 0.8825 - v
Epoch 35/50
278612/278612 [==============================] - 27s 97us/step - loss: 0.4562 - acc: 0.8820 - v
Epoch 36/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4453 - acc: 0.8845 - v
Epoch 37/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4437 - acc: 0.8846 - v
Epoch 38/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4375 - acc: 0.8864 - v
Epoch 39/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4348 - acc: 0.8871 - v
Epoch 40/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4325 - acc: 0.8881 - v
Epoch 41/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4268 - acc: 0.8894 - v
Epoch 42/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4242 - acc: 0.8895 - v
Epoch 43/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4206 - acc: 0.8909 - v
Epoch 44/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4206 - acc: 0.8910 - v
Epoch 45/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4218 - acc: 0.8905 - v
Epoch 46/50
278612/278612 [==============================] - 27s 98us/step - loss: 0.4125 - acc: 0.8926 - v
Epoch 47/50
278612/278612 [==============================] - 27s 95us/step - loss: 0.4096 - acc: 0.8929 - v
Epoch 48/50
278612/278612 [==============================] - 103s 369us/step - loss: 0.4084 - acc: 0.8942 -
Epoch 49/50
278612/278612 [==============================] - 26s 94us/step - loss: 0.4055 - acc: 0.8945 - v
Epoch 50/50
278612/278612 [==============================] - 26s 95us/step - loss: 0.4037 - acc: 0.8945 - v
```

5. Estimate an RNN with an LSTM layer

In [60]:
```
from keras.layers import LSTM

lstm = Sequential()
lstm.add(Embedding(10000, 25, input_length=100))
lstm.add(LSTM(25))
lstm.add(Dense(24, activation='softmax'))
```

```
lstm.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy
result_lstm = lstm.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs=50
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/50
278612/278612 [==============================] - 4093s 15ms/step - loss: 2.3026 - acc: 0.3026 -
Epoch 2/50
278612/278612 [==============================] - 75s 270us/step - loss: 1.3818 - acc: 0.6260 -
Epoch 3/50
278612/278612 [==============================] - 73s 263us/step - loss: 0.9714 - acc: 0.7564 -
Epoch 4/50
278612/278612 [==============================] - 71s 254us/step - loss: 0.7894 - acc: 0.8037 -
Epoch 5/50
278612/278612 [==============================] - 70s 251us/step - loss: 0.6971 - acc: 0.8234 -
Epoch 6/50
278612/278612 [==============================] - 71s 254us/step - loss: 0.6402 - acc: 0.8353 -
Epoch 7/50
278612/278612 [==============================] - 75s 268us/step - loss: 0.6006 - acc: 0.8441 -
Epoch 8/50
278612/278612 [==============================] - 73s 264us/step - loss: 0.5693 - acc: 0.8504 -
Epoch 9/50
278612/278612 [==============================] - 75s 270us/step - loss: 0.5440 - acc: 0.8558 -
Epoch 10/50
278612/278612 [==============================] - 73s 260us/step - loss: 0.5237 - acc: 0.8595 -
Epoch 11/50
278612/278612 [==============================] - 74s 265us/step - loss: 0.5062 - acc: 0.8635 -
Epoch 12/50
278612/278612 [==============================] - 73s 261us/step - loss: 0.4917 - acc: 0.8663 -
Epoch 13/50
278612/278612 [==============================] - 73s 262us/step - loss: 0.4790 - acc: 0.8693 -
Epoch 14/50
278612/278612 [==============================] - 71s 257us/step - loss: 0.4687 - acc: 0.8715 -
Epoch 15/50
278612/278612 [==============================] - 71s 256us/step - loss: 0.4582 - acc: 0.8741 -
Epoch 16/50
278612/278612 [==============================] - 73s 261us/step - loss: 0.4495 - acc: 0.8758 -
Epoch 17/50
278612/278612 [==============================] - 74s 266us/step - loss: 0.4411 - acc: 0.8778 -
Epoch 18/50
278612/278612 [==============================] - 73s 263us/step - loss: 0.4343 - acc: 0.8796 -
Epoch 19/50
278612/278612 [==============================] - 77s 276us/step - loss: 0.4279 - acc: 0.8809 -
Epoch 20/50
278612/278612 [==============================] - 76s 272us/step - loss: 0.4216 - acc: 0.8825 -
Epoch 21/50
278612/278612 [==============================] - 73s 261us/step - loss: 0.4156 - acc: 0.8839 -
Epoch 22/50
278612/278612 [==============================] - 72s 259us/step - loss: 0.4102 - acc: 0.8857 -
```

```
Epoch 23/50
278612/278612 [==============================] - 72s 260us/step - loss: 0.4049 - acc: 0.8869 -
Epoch 24/50
278612/278612 [==============================] - 73s 263us/step - loss: 0.4002 - acc: 0.8876 -
Epoch 25/50
278612/278612 [==============================] - 74s 265us/step - loss: 0.3955 - acc: 0.8889 -
Epoch 26/50
278612/278612 [==============================] - 73s 260us/step - loss: 0.3910 - acc: 0.8899 -
Epoch 27/50
278612/278612 [==============================] - 74s 267us/step - loss: 0.3873 - acc: 0.8911 -
Epoch 28/50
278612/278612 [==============================] - 74s 266us/step - loss: 0.3833 - acc: 0.8922 -
Epoch 29/50
278612/278612 [==============================] - 75s 270us/step - loss: 0.3792 - acc: 0.8929 -
Epoch 30/50
278612/278612 [==============================] - 75s 269us/step - loss: 0.3756 - acc: 0.8941 -
Epoch 31/50
278612/278612 [==============================] - 76s 271us/step - loss: 0.3723 - acc: 0.8950 -
Epoch 32/50
278612/278612 [==============================] - 76s 273us/step - loss: 0.3693 - acc: 0.8956 -
Epoch 33/50
278612/278612 [==============================] - 76s 274us/step - loss: 0.3653 - acc: 0.8969 -
Epoch 34/50
278612/278612 [==============================] - 76s 274us/step - loss: 0.3627 - acc: 0.8974 -
Epoch 35/50
278612/278612 [==============================] - 76s 271us/step - loss: 0.3595 - acc: 0.8982 -
Epoch 36/50
278612/278612 [==============================] - 73s 263us/step - loss: 0.3569 - acc: 0.8992 -
Epoch 37/50
278612/278612 [==============================] - 73s 260us/step - loss: 0.3540 - acc: 0.8999 -
Epoch 38/50
278612/278612 [==============================] - 72s 257us/step - loss: 0.3513 - acc: 0.9006 -
Epoch 39/50
278612/278612 [==============================] - 73s 260us/step - loss: 0.3491 - acc: 0.9011 -
Epoch 40/50
278612/278612 [==============================] - 75s 269us/step - loss: 0.3465 - acc: 0.9018 -
Epoch 41/50
278612/278612 [==============================] - 76s 273us/step - loss: 0.3439 - acc: 0.9029 -
Epoch 42/50
278612/278612 [==============================] - 76s 273us/step - loss: 0.3417 - acc: 0.9034 -
Epoch 43/50
278612/278612 [==============================] - 73s 263us/step - loss: 0.3391 - acc: 0.9039 -
Epoch 44/50
278612/278612 [==============================] - 74s 267us/step - loss: 0.3367 - acc: 0.9048 -
Epoch 45/50
278612/278612 [==============================] - 74s 267us/step - loss: 0.3350 - acc: 0.9048 -
Epoch 46/50
278612/278612 [==============================] - 75s 269us/step - loss: 0.3324 - acc: 0.9062 -
```

```
Epoch 47/50
278612/278612 [==============================] - 77s 275us/step - loss: 0.3306 - acc: 0.9064 -
Epoch 48/50
278612/278612 [==============================] - 77s 275us/step - loss: 0.3291 - acc: 0.9067 -
Epoch 49/50
278612/278612 [==============================] - 75s 268us/step - loss: 0.3265 - acc: 0.9072 -
Epoch 50/50
278612/278612 [==============================] - 76s 272us/step - loss: 0.3249 - acc: 0.9078 -
```

6. Estimate an RNN with a GRU layer

```
In [61]: from keras.layers import GRU

         gru = Sequential()
         gru.add(Embedding(10000, 25, input_length=100))
         gru.add(GRU(25))
         gru.add(Dense(24, activation='softmax'))
         gru.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']
         result_gru = gru.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs=50, b
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/50
278612/278612 [==============================] - 66s 237us/step - loss: 2.4846 - acc: 0.2287 -
Epoch 2/50
278612/278612 [==============================] - 63s 227us/step - loss: 1.7275 - acc: 0.5006 -
Epoch 3/50
278612/278612 [==============================] - 65s 233us/step - loss: 1.2263 - acc: 0.6657 -
Epoch 4/50
278612/278612 [==============================] - 67s 242us/step - loss: 0.9589 - acc: 0.7530 -
Epoch 5/50
278612/278612 [==============================] - 65s 234us/step - loss: 0.7906 - acc: 0.7994 -
Epoch 6/50
278612/278612 [==============================] - 178s 640us/step - loss: 0.6993 - acc: 0.8197 -
Epoch 7/50
278612/278612 [==============================] - 4784s 17ms/step - loss: 0.6409 - acc: 0.8329 -
Epoch 8/50
278612/278612 [==============================] - 63s 226us/step - loss: 0.5997 - acc: 0.8416 -
Epoch 9/50
278612/278612 [==============================] - 63s 226us/step - loss: 0.5680 - acc: 0.8484 -
Epoch 10/50
278612/278612 [==============================] - 63s 225us/step - loss: 0.5422 - acc: 0.8540 -
Epoch 11/50
278612/278612 [==============================] - 64s 229us/step - loss: 0.5208 - acc: 0.8591 -
Epoch 12/50
278612/278612 [==============================] - 70s 252us/step - loss: 0.5033 - acc: 0.8631 -
Epoch 13/50
278612/278612 [==============================] - 71s 254us/step - loss: 0.4882 - acc: 0.8666 -
```

```
Epoch 14/50
278612/278612 [==============================] - 73s 262us/step - loss: 0.4749 - acc: 0.8695 -
Epoch 15/50
278612/278612 [==============================] - 75s 269us/step - loss: 0.4635 - acc: 0.8720 -
Epoch 16/50
278612/278612 [==============================] - 66s 238us/step - loss: 0.4530 - acc: 0.8745 -
Epoch 17/50
278612/278612 [==============================] - 64s 229us/step - loss: 0.4437 - acc: 0.8769 -
Epoch 18/50
278612/278612 [==============================] - 63s 227us/step - loss: 0.4351 - acc: 0.8791 -
Epoch 19/50
278612/278612 [==============================] - 64s 231us/step - loss: 0.4274 - acc: 0.8808 -
Epoch 20/50
278612/278612 [==============================] - 66s 235us/step - loss: 0.4198 - acc: 0.8828 -
Epoch 21/50
278612/278612 [==============================] - 64s 230us/step - loss: 0.4129 - acc: 0.8847 -
Epoch 22/50
278612/278612 [==============================] - 66s 236us/step - loss: 0.4065 - acc: 0.8864 -
Epoch 23/50
278612/278612 [==============================] - 64s 231us/step - loss: 0.4006 - acc: 0.8880 -
Epoch 24/50
278612/278612 [==============================] - 65s 234us/step - loss: 0.3949 - acc: 0.8898 -
Epoch 25/50
278612/278612 [==============================] - 66s 238us/step - loss: 0.3893 - acc: 0.8909 -
Epoch 26/50
278612/278612 [==============================] - 63s 226us/step - loss: 0.3841 - acc: 0.8928 -
Epoch 27/50
278612/278612 [==============================] - 63s 226us/step - loss: 0.3790 - acc: 0.8939 -
Epoch 28/50
278612/278612 [==============================] - 62s 222us/step - loss: 0.3743 - acc: 0.8951 -
Epoch 29/50
278612/278612 [==============================] - 63s 225us/step - loss: 0.3699 - acc: 0.8964 -
Epoch 30/50
278612/278612 [==============================] - 64s 231us/step - loss: 0.3659 - acc: 0.8978 -
Epoch 31/50
278612/278612 [==============================] - 64s 230us/step - loss: 0.3612 - acc: 0.8989 -
Epoch 32/50
278612/278612 [==============================] - 62s 222us/step - loss: 0.3570 - acc: 0.9000 -
Epoch 33/50
278612/278612 [==============================] - 63s 226us/step - loss: 0.3533 - acc: 0.9011 -
Epoch 34/50
278612/278612 [==============================] - 62s 223us/step - loss: 0.3495 - acc: 0.9019 -
Epoch 35/50
278612/278612 [==============================] - 65s 234us/step - loss: 0.3458 - acc: 0.9034 -
Epoch 36/50
278612/278612 [==============================] - 66s 236us/step - loss: 0.3424 - acc: 0.9042 -
Epoch 37/50
278612/278612 [==============================] - 65s 234us/step - loss: 0.3387 - acc: 0.9048 -
```

```
Epoch 38/50
278612/278612 [==============================] - 67s 241us/step - loss: 0.3352 - acc: 0.9062 -
Epoch 39/50
278612/278612 [==============================] - 67s 241us/step - loss: 0.3323 - acc: 0.9068 -
Epoch 40/50
278612/278612 [==============================] - 65s 232us/step - loss: 0.3289 - acc: 0.9077 -
Epoch 41/50
278612/278612 [==============================] - 65s 235us/step - loss: 0.3261 - acc: 0.9088 -
Epoch 42/50
278612/278612 [==============================] - 66s 237us/step - loss: 0.3233 - acc: 0.9096 -
Epoch 43/50
278612/278612 [==============================] - 65s 235us/step - loss: 0.3206 - acc: 0.9102 -
Epoch 44/50
278612/278612 [==============================] - 66s 238us/step - loss: 0.3177 - acc: 0.9111 -
Epoch 45/50
278612/278612 [==============================] - 68s 245us/step - loss: 0.3152 - acc: 0.9119 -
Epoch 46/50
278612/278612 [==============================] - 67s 241us/step - loss: 0.3127 - acc: 0.9126 -
Epoch 47/50
278612/278612 [==============================] - 66s 238us/step - loss: 0.3099 - acc: 0.9132 -
Epoch 48/50
278612/278612 [==============================] - 66s 237us/step - loss: 0.3076 - acc: 0.9141 -
Epoch 49/50
278612/278612 [==============================] - 66s 237us/step - loss: 0.3051 - acc: 0.9145 -
Epoch 50/50
278612/278612 [==============================] - 65s 232us/step - loss: 0.3028 - acc: 0.9149 -
```

7. Estimate five additional neural network models with different configurations of hyperparameters

```
In [63]: model1 = Sequential()
         model1.add(Embedding(10000, 25, input_length=100))
         model1.add(SimpleRNN(25, return_sequences=True))
         model1.add(SimpleRNN(25))
         model1.add(Dense(24, activation='softmax'))
         model1.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accura
         result_model1 = model1.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/25
278612/278612 [==============================] - 57s 204us/step - loss: 2.1274 - acc: 0.3818 -
Epoch 2/25
278612/278612 [==============================] - 56s 200us/step - loss: 1.3324 - acc: 0.6418 -
Epoch 3/25
278612/278612 [==============================] - 54s 193us/step - loss: 1.0594 - acc: 0.7223 -
Epoch 4/25
278612/278612 [==============================] - 55s 198us/step - loss: 0.9109 - acc: 0.7634 -
Epoch 5/25
```

```
278612/278612 [==============================] - 55s 197us/step - loss: 0.8176 - acc: 0.7887 -
Epoch 6/25
278612/278612 [==============================] - 56s 201us/step - loss: 0.7588 - acc: 0.8037 -
Epoch 7/25
278612/278612 [==============================] - 57s 206us/step - loss: 0.7201 - acc: 0.8135 -
Epoch 8/25
278612/278612 [==============================] - 57s 205us/step - loss: 0.6829 - acc: 0.8233 -
Epoch 9/25
278612/278612 [==============================] - 56s 202us/step - loss: 0.6546 - acc: 0.8297 -
Epoch 10/25
278612/278612 [==============================] - 57s 206us/step - loss: 0.6300 - acc: 0.8369 -
Epoch 11/25
278612/278612 [==============================] - 56s 200us/step - loss: 0.6099 - acc: 0.8415 -
Epoch 12/25
278612/278612 [==============================] - 56s 200us/step - loss: 0.5907 - acc: 0.8463 -
Epoch 13/25
278612/278612 [==============================] - 54s 194us/step - loss: 0.5772 - acc: 0.8503 -
Epoch 14/25
278612/278612 [==============================] - 54s 195us/step - loss: 0.5616 - acc: 0.8539 -
Epoch 15/25
278612/278612 [==============================] - 54s 193us/step - loss: 0.5492 - acc: 0.8571 -
Epoch 16/25
278612/278612 [==============================] - 54s 194us/step - loss: 0.5369 - acc: 0.8603 -
Epoch 17/25
278612/278612 [==============================] - 54s 193us/step - loss: 0.5264 - acc: 0.8630 -
Epoch 18/25
278612/278612 [==============================] - 54s 194us/step - loss: 0.5191 - acc: 0.8651 -
Epoch 19/25
278612/278612 [==============================] - 55s 197us/step - loss: 0.5082 - acc: 0.8675 -
Epoch 20/25
278612/278612 [==============================] - 55s 199us/step - loss: 0.5005 - acc: 0.8696 -
Epoch 21/25
278612/278612 [==============================] - 55s 198us/step - loss: 0.4927 - acc: 0.8718 -
Epoch 22/25
278612/278612 [==============================] - 56s 200us/step - loss: 0.4851 - acc: 0.8735 -
Epoch 23/25
278612/278612 [==============================] - 54s 195us/step - loss: 0.4827 - acc: 0.8741 -
Epoch 24/25
278612/278612 [==============================] - 54s 194us/step - loss: 0.4717 - acc: 0.8770 -
Epoch 25/25
278612/278612 [==============================] - 57s 206us/step - loss: 0.4673 - acc: 0.8779 -
```

In [68]:
```
model2 = Sequential()
model2.add(Embedding(10000, 25, input_length=100))
model2.add(LSTM(25, return_sequences=True))
model2.add(LSTM(25))
model2.add(Dense(24, activation='softmax'))
```

```
model2.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accura
result_model2 = model2.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/25
278612/278612 [==============================] - 154s 552us/step - loss: 2.2230 - acc: 0.3419 -
Epoch 2/25
278612/278612 [==============================] - 162s 581us/step - loss: 1.4074 - acc: 0.6121 -
Epoch 3/25
278612/278612 [==============================] - 148s 533us/step - loss: 1.0517 - acc: 0.7258 -
Epoch 4/25
278612/278612 [==============================] - 152s 544us/step - loss: 0.8625 - acc: 0.7798 -
Epoch 5/25
278612/278612 [==============================] - 153s 551us/step - loss: 0.7498 - acc: 0.8086 -
Epoch 6/25
278612/278612 [==============================] - 149s 534us/step - loss: 0.6772 - acc: 0.8252 -
Epoch 7/25
278612/278612 [==============================] - 147s 529us/step - loss: 0.6269 - acc: 0.8356 -
Epoch 8/25
278612/278612 [==============================] - 147s 527us/step - loss: 0.5891 - acc: 0.8435 -
Epoch 9/25
278612/278612 [==============================] - 147s 527us/step - loss: 0.5601 - acc: 0.8498 -
Epoch 10/25
278612/278612 [==============================] - 147s 528us/step - loss: 0.5377 - acc: 0.8551 -
Epoch 11/25
278612/278612 [==============================] - 147s 526us/step - loss: 0.5194 - acc: 0.8587 -
Epoch 12/25
278612/278612 [==============================] - 148s 532us/step - loss: 0.5041 - acc: 0.8625 -
Epoch 13/25
278612/278612 [==============================] - 149s 534us/step - loss: 0.4902 - acc: 0.8655 -
Epoch 14/25
278612/278612 [==============================] - 145s 520us/step - loss: 0.4781 - acc: 0.8680 -
Epoch 15/25
278612/278612 [==============================] - 147s 528us/step - loss: 0.4672 - acc: 0.8710 -
Epoch 16/25
278612/278612 [==============================] - 149s 535us/step - loss: 0.4572 - acc: 0.8731 -
Epoch 17/25
278612/278612 [==============================] - 149s 535us/step - loss: 0.4481 - acc: 0.8756 -
Epoch 18/25
278612/278612 [==============================] - 148s 530us/step - loss: 0.4392 - acc: 0.8780 -
Epoch 19/25
278612/278612 [==============================] - 406s 1ms/step - loss: 0.4324 - acc: 0.8796 - v
Epoch 20/25
278612/278612 [==============================] - 147s 529us/step - loss: 0.4250 - acc: 0.8813 -
Epoch 21/25
278612/278612 [==============================] - 144s 517us/step - loss: 0.4187 - acc: 0.8829 -
Epoch 22/25
278612/278612 [==============================] - 147s 526us/step - loss: 0.4123 - acc: 0.8847 -
```

```
Epoch 23/25
278612/278612 [==============================] - 143s 513us/step - loss: 0.4063 - acc: 0.8859 -
Epoch 24/25
278612/278612 [==============================] - 144s 517us/step - loss: 0.4006 - acc: 0.8875 -
Epoch 25/25
278612/278612 [==============================] - 145s 519us/step - loss: 0.3959 - acc: 0.8888 -
```

In [71]:
```
model3 = Sequential()
model3.add(Embedding(10000, 25, input_length=100))
model3.add(SimpleRNN(25, dropout=0.3))
model3.add(Dense(24, activation='softmax'))
model3.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accurac
result_model3 = model3.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/25
278612/278612 [==============================] - 30s 107us/step - loss: 2.3317 - acc: 0.3014 -
Epoch 2/25
278612/278612 [==============================] - 29s 102us/step - loss: 1.6292 - acc: 0.5322 -
Epoch 3/25
278612/278612 [==============================] - 29s 103us/step - loss: 1.3952 - acc: 0.6128 -
Epoch 4/25
278612/278612 [==============================] - 29s 103us/step - loss: 1.2855 - acc: 0.6475 -
Epoch 5/25
278612/278612 [==============================] - 29s 104us/step - loss: 1.2151 - acc: 0.6697 -
Epoch 6/25
278612/278612 [==============================] - 29s 105us/step - loss: 1.1578 - acc: 0.6874 -
Epoch 7/25
278612/278612 [==============================] - 29s 105us/step - loss: 1.1030 - acc: 0.7053 -
Epoch 8/25
278612/278612 [==============================] - 30s 108us/step - loss: 1.0548 - acc: 0.7185 -
Epoch 9/25
278612/278612 [==============================] - 30s 109us/step - loss: 1.0140 - acc: 0.7323 -
Epoch 10/25
278612/278612 [==============================] - 30s 109us/step - loss: 0.9803 - acc: 0.7422 -
Epoch 11/25
278612/278612 [==============================] - 30s 108us/step - loss: 0.9471 - acc: 0.7517 -
Epoch 12/25
278612/278612 [==============================] - 30s 109us/step - loss: 0.9167 - acc: 0.7607 -
Epoch 13/25
278612/278612 [==============================] - 30s 108us/step - loss: 0.8893 - acc: 0.7678 -
Epoch 14/25
278612/278612 [==============================] - 29s 103us/step - loss: 0.8652 - acc: 0.7739 -
Epoch 15/25
278612/278612 [==============================] - 29s 106us/step - loss: 0.8454 - acc: 0.7799 -
Epoch 16/25
278612/278612 [==============================] - 30s 107us/step - loss: 0.8264 - acc: 0.7843 -
```

```
Epoch 17/25
278612/278612 [==============================] - 29s 103us/step - loss: 0.8101 - acc: 0.7901 -
Epoch 18/25
278612/278612 [==============================] - 29s 104us/step - loss: 0.7957 - acc: 0.7938 -
Epoch 19/25
278612/278612 [==============================] - 29s 105us/step - loss: 0.7856 - acc: 0.7964 -
Epoch 20/25
278612/278612 [==============================] - 38s 136us/step - loss: 0.7754 - acc: 0.7990 -
Epoch 21/25
278612/278612 [==============================] - 28s 102us/step - loss: 0.7671 - acc: 0.8014 -
Epoch 22/25
278612/278612 [==============================] - 28s 102us/step - loss: 0.7617 - acc: 0.8024 -
Epoch 23/25
278612/278612 [==============================] - 29s 103us/step - loss: 0.7557 - acc: 0.8039 -
Epoch 24/25
278612/278612 [==============================] - 28s 101us/step - loss: 0.7485 - acc: 0.8057 -
Epoch 25/25
278612/278612 [==============================] - 28s 102us/step - loss: 0.7432 - acc: 0.8074 -
```

In [73]:
```
model4 = Sequential()
model4.add(Embedding(10000, 25, input_length=100))
model4.add(LSTM(25, dropout=0.3))
model4.add(Dense(24, activation='softmax'))
model4.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accurac
result_model4 = model4.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/25
278612/278612 [==============================] - 78s 280us/step - loss: 2.2343 - acc: 0.3404 -
Epoch 2/25
278612/278612 [==============================] - 77s 277us/step - loss: 1.3589 - acc: 0.6303 -
Epoch 3/25
278612/278612 [==============================] - 77s 276us/step - loss: 0.9963 - acc: 0.7463 -
Epoch 4/25
278612/278612 [==============================] - 78s 279us/step - loss: 0.8409 - acc: 0.7889 -
Epoch 5/25
278612/278612 [==============================] - 78s 281us/step - loss: 0.7610 - acc: 0.8064 -
Epoch 6/25
278612/278612 [==============================] - 79s 282us/step - loss: 0.7108 - acc: 0.8176 -
Epoch 7/25
278612/278612 [==============================] - 78s 281us/step - loss: 0.6763 - acc: 0.8244 -
Epoch 8/25
278612/278612 [==============================] - 78s 281us/step - loss: 0.6464 - acc: 0.8311 -
Epoch 9/25
278612/278612 [==============================] - 79s 285us/step - loss: 0.6250 - acc: 0.8354 -
Epoch 10/25
278612/278612 [==============================] - 78s 281us/step - loss: 0.6057 - acc: 0.8391 -
```

```
Epoch 11/25
278612/278612 [==============================] - 80s 286us/step - loss: 0.5885 - acc: 0.8429 -
Epoch 12/25
278612/278612 [==============================] - 86s 309us/step - loss: 0.5750 - acc: 0.8454 -
Epoch 13/25
278612/278612 [==============================] - 85s 307us/step - loss: 0.5622 - acc: 0.8484 -
Epoch 14/25
278612/278612 [==============================] - 85s 305us/step - loss: 0.5513 - acc: 0.8496 -
Epoch 15/25
278612/278612 [==============================] - 85s 305us/step - loss: 0.5408 - acc: 0.8516 -
Epoch 16/25
278612/278612 [==============================] - 85s 306us/step - loss: 0.5309 - acc: 0.8537 -
Epoch 17/25
278612/278612 [==============================] - 85s 306us/step - loss: 0.5238 - acc: 0.8555 -
Epoch 18/25
278612/278612 [==============================] - 82s 294us/step - loss: 0.5170 - acc: 0.8567 -
Epoch 19/25
278612/278612 [==============================] - 84s 302us/step - loss: 0.5107 - acc: 0.8579 -
Epoch 20/25
278612/278612 [==============================] - 84s 302us/step - loss: 0.5047 - acc: 0.8590 -
Epoch 21/25
278612/278612 [==============================] - 80s 286us/step - loss: 0.4993 - acc: 0.8602 -
Epoch 22/25
278612/278612 [==============================] - 83s 298us/step - loss: 0.4949 - acc: 0.8614 -
Epoch 23/25
278612/278612 [==============================] - 81s 289us/step - loss: 0.4916 - acc: 0.8624 -
Epoch 24/25
278612/278612 [==============================] - 80s 287us/step - loss: 0.4870 - acc: 0.8633 -
Epoch 25/25
278612/278612 [==============================] - 86s 310us/step - loss: 0.4838 - acc: 0.8641 -
```

```
In [75]: model5 = Sequential()
         model5.add(Embedding(10000, 25, input_length=100))
         model5.add(GRU(25, dropout=0.3))
         model5.add(Dense(24, activation='softmax'))
         model5.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accurac
         result_model5 = model5.fit(train_x, train_y, validation_data=(valid_x,valid_y), epochs
```

```
Train on 278612 samples, validate on 69649 samples
Epoch 1/25
278612/278612 [==============================] - 85s 304us/step - loss: 2.4753 - acc: 0.2468 -
Epoch 2/25
278612/278612 [==============================] - 82s 294us/step - loss: 1.6597 - acc: 0.5441 -
Epoch 3/25
278612/278612 [==============================] - 78s 281us/step - loss: 1.1924 - acc: 0.6903 -
Epoch 4/25
278612/278612 [==============================] - 77s 276us/step - loss: 0.9666 - acc: 0.7568 -
```

```
Epoch 5/25
278612/278612 [==============================] - 84s 302us/step - loss: 0.8595 - acc: 0.7841 -
Epoch 6/25
278612/278612 [==============================] - 84s 303us/step - loss: 0.7888 - acc: 0.8002 -
Epoch 7/25
278612/278612 [==============================] - 84s 300us/step - loss: 0.7395 - acc: 0.8098 -
Epoch 8/25
278612/278612 [==============================] - 82s 296us/step - loss: 0.7018 - acc: 0.8167 -
Epoch 9/25
278612/278612 [==============================] - 79s 284us/step - loss: 0.6715 - acc: 0.8235 -
Epoch 10/25
278612/278612 [==============================] - 81s 291us/step - loss: 0.6479 - acc: 0.8280 -
Epoch 11/25
278612/278612 [==============================] - 83s 297us/step - loss: 0.6286 - acc: 0.8314 -
Epoch 12/25
278612/278612 [==============================] - 86s 310us/step - loss: 0.6125 - acc: 0.8347 -
Epoch 13/25
278612/278612 [==============================] - 88s 317us/step - loss: 0.5972 - acc: 0.8376 -
Epoch 14/25
278612/278612 [==============================] - 81s 291us/step - loss: 0.5842 - acc: 0.8401 -
Epoch 15/25
278612/278612 [==============================] - 80s 287us/step - loss: 0.5707 - acc: 0.8429 -
Epoch 16/25
278612/278612 [==============================] - 77s 276us/step - loss: 0.5630 - acc: 0.8442 -
Epoch 17/25
278612/278612 [==============================] - 74s 266us/step - loss: 0.5521 - acc: 0.8462 -
Epoch 18/25
278612/278612 [==============================] - 73s 263us/step - loss: 0.5443 - acc: 0.8486 -
Epoch 19/25
278612/278612 [==============================] - 74s 265us/step - loss: 0.5362 - acc: 0.8501 -
Epoch 20/25
278612/278612 [==============================] - 84s 301us/step - loss: 0.5313 - acc: 0.8511 -
Epoch 21/25
278612/278612 [==============================] - 84s 302us/step - loss: 0.5239 - acc: 0.8533 -
Epoch 22/25
278612/278612 [==============================] - 84s 300us/step - loss: 0.5189 - acc: 0.8544 -
Epoch 23/25
278612/278612 [==============================] - 84s 302us/step - loss: 0.5139 - acc: 0.8553 -
Epoch 24/25
278612/278612 [==============================] - 76s 274us/step - loss: 0.5096 - acc: 0.8561 -
Epoch 25/25
278612/278612 [==============================] - 74s 266us/step - loss: 0.5058 - acc: 0.8571 -
```

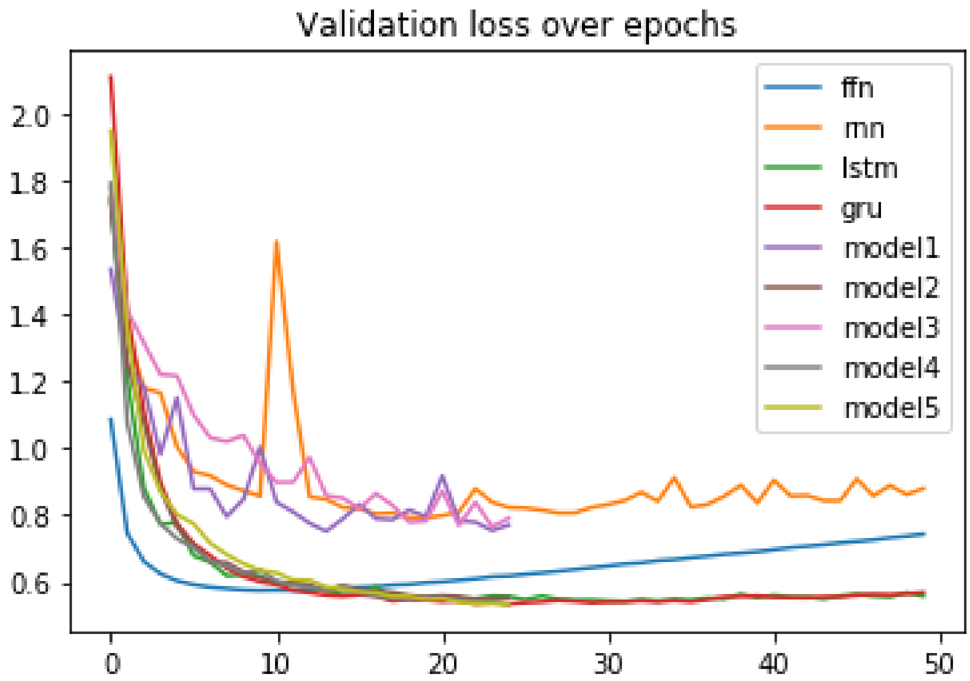
8.For each model, plot the validation loss and accuracy over each epoch.

```
In [77]: import matplotlib.pyplot as plt
```

```
In [82]: models_list = [result_ffn, result_rnn, result_lstm, result_gru, result_model1, result_
```

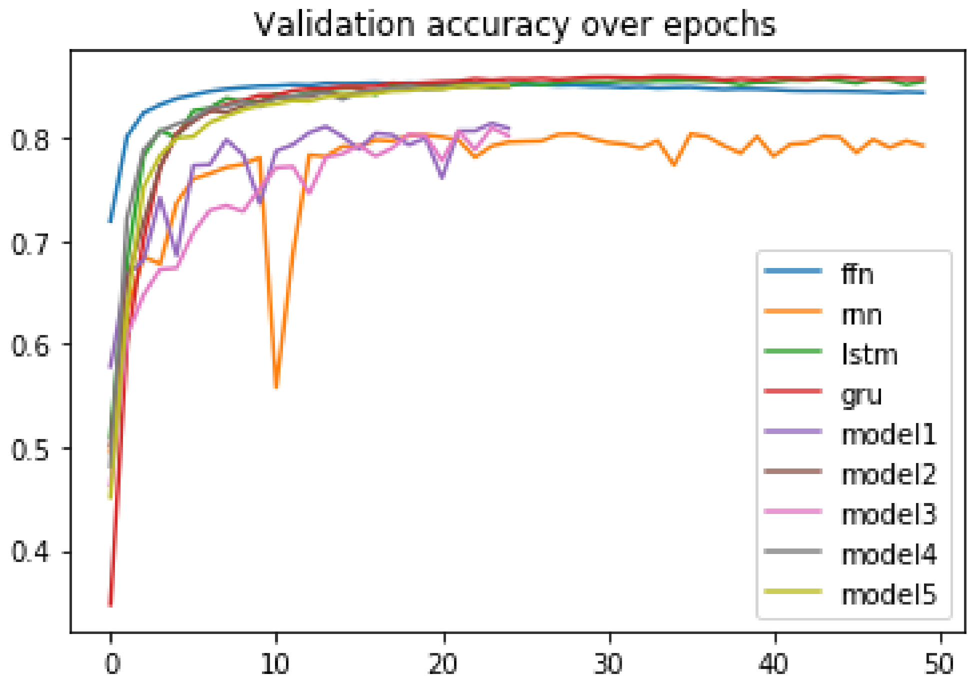
```
for model in models_list:
    plt.plot(model.history['val_loss'])

plt.legend(['ffn', 'rnn', 'lstm', 'gru', 'model1', 'model2', 'model3', 'model4', 'mode
plt.title('Validation loss over epochs')
plt.show()
```




```
In [83]: for model in models_list:
             plt.plot(model.history['val_acc'])

         plt.legend(['ffn', 'rnn', 'lstm', 'gru', 'model1', 'model2', 'model3', 'model4', 'mode
         plt.title('Validation accuracy over epochs')
         plt.show()
```



9.Select the best performing model based on the validation set and evaluate its performance using the test set. Assume that with hand-coding we can achieve a 95% accuracy rate. Would your neural network perform better or worse than hand-coding?

From above two plots we can see that, the best performing model is the model with GRU layer because it seems to have the highest validation accuracy and lowest validation loss.

```
In [84]: gru.evaluate(test_x, test_y)

69649/69649 [==============================] - 14s 203us/step


Out[84]: [0.5703272305414112, 0.8573848870826176]
```

The test accuracy rate using GRU model is 85.7%, it performs worse than hand-coding.

```
In [ ]:
```