```
from sklearn.feature_extraction import DictVectorizer
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.preprocessing import LabelEncoder
from google.colab import drive
import json
from sklearn.metrics import accuracy_score
from keras.models import Sequential
from keras.layers import Dense, Dropout, BatchNormalization
from tensorflow.keras.utils import to_categorical
```

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
actual_tags = []
with open ('/content/drive/MyDrive/Cleaned_Sentences_Task/tags_original.txt', 'r') as file:
    actual_tags = json.load(file)
```

```
removed = []
with open ('/content/drive/MyDrive/Cleaned_Sentences_Task/sentences_without_tags.txt', 'r') as file:
    removed = json.load(file)
```

```
print(len(removed))
```

```
331364
```

```
def extract_features(sentence, index):
  return {
      'word':sentence[index],
      'is_first':index==0,
      'is_last':index ==len(sentence)-1,
      'prefix-1':sentence[index][0],
      'prefix-2':sentence[index][:2],
      'prefix-3':sentence[index][:3],
      'prefix-3':sentence[index][:4],
```

```
      'suffix-1':sentence[index][-1],
      'suffix-2':sentence[index][-2:],
      'suffix-3':sentence[index][-3:],
      'suffix-3':sentence[index][-4:],
      'prev_word':'' if index == 0 else sentence[index-1],
      'next_word':'' if index == 1 else sentence[index+1],
      'has_hyphen': '-' in sentence[index],
      'is_numeric': sentence[index].isdigit()
  }
```

```
def transform_to_dataset(sentences):
  X, y = [], []
  for sents in sentences:
    for index in range(len(sents)):
      X.append(extract_features(sents, index))
  return X, actual_tags[0: 22000]
```

```
X_, y_ = transform_to_dataset(removed[0: 11000])
for i in range(0, 100):
  print(X_[i], " ----------> ", y_[i])
```

```
{'word': 'आग', 'is_first': True, 'is_last': False, 'prefix-1': 'आ', 'prefix-2': 'आग', 'prefix-3': 'आग', 'suffix-1': 'ग',
{'word': 'की', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'की', 'prefix-3': 'की', 'suffix-1': 'ी', 's
{'word': 'लिए', 'is_first': True, 'is_last': False, 'prefix-1': 'ल', 'prefix-2': 'लि', 'prefix-3': 'लिए', 'suffix-1': 'ए', '
{'word': 'सांडिआ', 'is_first': False, 'is_last': True, 'prefix-1': 'स', 'prefix-2': 'सा', 'prefix-3': 'सांड', 'suffix-1': 'आ
{'word': 'की', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'की', 'prefix-3': 'की', 'suffix-1': 'ी', 's
{'word': 'किरण', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'कि', 'prefix-3': 'किरण', 'suffix-1': 'ण
{'word': 'जानी', 'is_first': True, 'is_last': False, 'prefix-1': 'ज', 'prefix-2': 'जा', 'prefix-3': 'जानी', 'suffix-1': 'ी',
{'word': 'चाहिए', 'is_first': False, 'is_last': True, 'prefix-1': 'च', 'prefix-2': 'चा', 'prefix-3': 'चाहि', 'suffix-1': 'ए',
{'word': 'बिहार', 'is_first': True, 'is_last': False, 'prefix-1': 'ब', 'prefix-2': 'बि', 'prefix-3': 'बिहा', 'suffix-1': 'र',
{'word': 'की', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'की', 'prefix-3': 'की', 'suffix-1': 'ी', 's
{'word': 'मोहम्मद', 'is_first': True, 'is_last': False, 'prefix-1': 'म', 'prefix-2': 'मो', 'prefix-3': 'मोहम', 'suffix-1': 'द
{'word': 'हकीम', 'is_first': False, 'is_last': True, 'prefix-1': 'ह', 'prefix-2': 'हक', 'prefix-3': 'हकीम', 'suffix-1': 'म
{'word': 'ने', 'is_first': True, 'is_last': False, 'prefix-1': 'न', 'prefix-2': 'ने', 'prefix-3': 'ने', 'suffix-1': 'े', 'suffi
{'word': 'बताया', 'is_first': False, 'is_last': True, 'prefix-1': 'ब', 'prefix-2': 'बत', 'prefix-3': 'बताय', 'suffix-1': 'ा',
{'word': 'को', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'को', 'prefix-3': 'को', 'suffix-1': 'ो', 's
{'word': 'खत्म', 'is_first': False, 'is_last': True, 'prefix-1': 'ख', 'prefix-2': 'खत', 'prefix-3': 'खत्म', 'suffix-1': 'म',
{'word': 'के', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'के', 'prefix-3': 'के', 'suffix-1': 'े', 'suf
{'word': 'मरने', 'is_first': False, 'is_last': True, 'prefix-1': 'म', 'prefix-2': 'मर', 'prefix-3': 'मरने', 'suffix-1': 'े', '
{'word': 'कभी', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'कभ', 'prefix-3': 'कभी', 'suffix-1': 'ी',
```

```
{'word': 'कुछ', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'कु', 'prefix-3': 'कुछ', 'suffix-1': 'छ',
{'word': 'लादेन', 'is_first': True, 'is_last': False, 'prefix-1': 'ल', 'prefix-2': 'ला', 'prefix-3': 'लादे', 'suffix-1': 'न',
{'word': 'की', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'की', 'prefix-3': 'की', 'suffix-1': 'ी', 's
{'word': 'रहे', 'is_first': True, 'is_last': False, 'prefix-1': 'र', 'prefix-2': 'रह', 'prefix-3': 'रहे', 'suffix-1': 'े', 'su
{'word': 'हैं', 'is_first': False, 'is_last': True, 'prefix-1': 'ह', 'prefix-2': 'है', 'prefix-3': 'हैं', 'suffix-1': 'ं', 'suffi
{'word': 'लिए', 'is_first': True, 'is_last': False, 'prefix-1': 'ल', 'prefix-2': 'लि', 'prefix-3': 'लिए', 'suffix-1': 'ए', '
{'word': 'बचाने', 'is_first': False, 'is_last': True, 'prefix-1': 'ब', 'prefix-2': 'बच', 'prefix-3': 'बचान', 'suffix-1': 'े',
{'word': 'कांग्रेस', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'का', 'prefix-3': 'कांग', 'suffix-1': 'स',
{'word': 'में', 'is_first': False, 'is_last': True, 'prefix-1': 'म', 'prefix-2': 'मे', 'prefix-3': 'में', 'suffix-1': 'ं', 'suff
{'word': 'युवाओं', 'is_first': True, 'is_last': False, 'prefix-1': 'य', 'prefix-2': 'यु', 'prefix-3': 'युवा', 'suffix-1': 'ं', '
{'word': 'की', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'की', 'prefix-3': 'की', 'suffix-1': 'ी', 's
{'word': 'जारी', 'is_first': True, 'is_last': False, 'prefix-1': 'ज', 'prefix-2': 'जा', 'prefix-3': 'जारी', 'suffix-1': 'ी',
{'word': 'करने', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'कर', 'prefix-3': 'करने', 'suffix-1': 'े',
{'word': 'मसले', 'is_first': True, 'is_last': False, 'prefix-1': 'म', 'prefix-2': 'मस', 'prefix-3': 'मसले', 'suffix-1': 'े',
{'word': 'को', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'को', 'prefix-3': 'को', 'suffix-1': 'ो', 's
{'word': 'के', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'के', 'prefix-3': 'के', 'suffix-1': 'े', 'suf
{'word': 'लोगों', 'is_first': False, 'is_last': True, 'prefix-1': 'ल', 'prefix-2': 'लो', 'prefix-3': 'लोगो', 'suffix-1': 'ं', '
{'word': 'रही', 'is_first': True, 'is_last': False, 'prefix-1': 'र', 'prefix-2': 'रह', 'prefix-3': 'रही', 'suffix-1': 'ी', 's
{'word': 'है', 'is_first': False, 'is_last': True, 'prefix-1': 'ह', 'prefix-2': 'है', 'prefix-3': 'है', 'suffix-1': 'ै', 'suffi
{'word': 'को', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'को', 'prefix-3': 'को', 'suffix-1': 'ो', 's
{'word': 'गहरा', 'is_first': False, 'is_last': True, 'prefix-1': 'ग', 'prefix-2': 'गह', 'prefix-3': 'गहरा', 'suffix-1': 'ा',
{'word': 'अभी', 'is_first': True, 'is_last': False, 'prefix-1': 'अ', 'prefix-2': 'अभ', 'prefix-3': 'अभी', 'suffix-1': 'ी',
{'word': 'इसका', 'is_first': False, 'is_last': True, 'prefix-1': 'इ', 'prefix-2': 'इस', 'prefix-3': 'इसका', 'suffix-1': 'ा'
{'word': 'फांसी', 'is_first': True, 'is_last': False, 'prefix-1': 'फ', 'prefix-2': 'फा', 'prefix-3': 'फांस', 'suffix-1': 'ी',
{'word': 'लग', 'is_first': False, 'is_last': True, 'prefix-1': 'ल', 'prefix-2': 'लग', 'prefix-3': 'लग', 'suffix-1': 'ग', '
{'word': 'चढ़त', 'is_first': True, 'is_last': False, 'prefix-1': 'च', 'prefix-2': 'चढ़', 'prefix-3': 'चढ़त', 'suffix-1': 'त',
{'word': 'के', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'के', 'prefix-3': 'के', 'suffix-1': 'े', 'suf
{'word': 'साथ', 'is_first': True, 'is_last': False, 'prefix-1': 'स', 'prefix-2': 'सा', 'prefix-3': 'साथ', 'suffix-1': 'थ', '
{'word': 'समूचे', 'is_first': False, 'is_last': True, 'prefix-1': 'स', 'prefix-2': 'सम', 'prefix-3': 'समूच', 'suffix-1': 'े',
{'word': 'घुसा', 'is_first': True, 'is_last': False, 'prefix-1': 'घ', 'prefix-2': 'घु', 'prefix-3': 'घुसा', 'suffix-1': 'ा', 's
{'word': 'NULL', 'is_first': False, 'is_last': True, 'prefix-1': 'N', 'prefix-2': 'NU', 'prefix-3': 'NULL', 'suffix-1': 'L',
{'word': 'सफल', 'is_first': True, 'is_last': False, 'prefix-1': 'स', 'prefix-2': 'सफ', 'prefix-3': 'सफल', 'suffix-1': 'ल
{'word': 'रहे', 'is_first': False, 'is_last': True, 'prefix-1': 'र', 'prefix-2': 'रह', 'prefix-3': 'रहे', 'suffix-1': 'े', 'su
{'word': '२६९९.२०', 'is_first': True, 'is_last': False, 'prefix-1': '२', 'prefix-2': '२६', 'prefix-3': '२६९९', 'suffix-1': '
{'word': 'करोड़', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'कर', 'prefix-3': 'करोड़', 'suffix-1': 'ड़
{'word': 'केंद्र', 'is_first': True, 'is_last': False, 'prefix-1': 'क', 'prefix-2': 'के', 'prefix-3': 'केंद', 'suffix-1': 'र', 's
{'word': 'के', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'के', 'prefix-3': 'के', 'suffix-1': 'े', 'suf
{'word': 'भारी', 'is_first': True, 'is_last': False, 'prefix-1': 'भ', 'prefix-2': 'भा', 'prefix-3': 'भारी', 'suffix-1': 'ी', '
{'word': 'क्षति', 'is_first': False, 'is_last': True, 'prefix-1': 'क', 'prefix-2': 'क', 'prefix-3': 'क्षत', 'suffix-1': 'ि', 's
```

```
1 print(len(X ), len(y ))
```

```
22000 22000
```

```
print(y_[0: 10])
print(actual_tags[0: 10])
```

```
['nn', 'psp', 'psp', 'nnpc', 'psp', 'nnpc', 'vaux', 'vaux', 'nnp', 'psp']
['nn', 'psp', 'psp', 'nnpc', 'psp', 'nnpc', 'vaux', 'vaux', 'nnp', 'psp']
```

```
X_train, X_test, y_train, y_test = train_test_split(X_, y_, train_size=0.75)
```

```
dict_vectorizer = DictVectorizer(sparse=False)
dict_vectorizer.fit(X_)
X_train = dict_vectorizer.transform(X_train)
X_test = dict_vectorizer.transform(X_test)
```

```
X_ = dict_vectorizer.transform(X_)
```

```
label_encoder = LabelEncoder()
label_encoder.fit(y_)
y_train = label_encoder.transform(y_train)
y_train_cat = to_categorical(y_train, num_classes=len(label_encoder.classes_))
```

```
model_deep = Sequential()
model_deep.add(Dense(1024, activation='relu'))
model_deep.add(Dropout(0.2))
model_deep.add(BatchNormalization())
model_deep.add(Dense(512, activation='relu'))
model_deep.add(Dropout(0.2))
model_deep.add(Dense(256, activation='relu'))
model_deep.add(Dropout(0.2))
model_deep.add(BatchNormalization())
model_deep.add(Dense(128, activation='relu'))
model_deep.add(Dropout(0.2))
model_deep.add(Dense(len(label_encoder.classes_), activation='softmax'))
```

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

```
model_deep.fit(X_train, y_train_cat, validation_split=0.2, epochs=50, batch_size=128)
```

```
Epoch 1/50
104/104 [==============================] - 31s 295ms/step - loss: 1.1723 - accuracy: 0.6736 - val_loss: 2.4328 - val_accurac
Epoch 2/50
104/104 [==============================] - 30s 290ms/step - loss: 0.4240 - accuracy: 0.8755 - val_loss: 2.1762 - val_accurac
Epoch 3/50
104/104 [==============================] - 30s 289ms/step - loss: 0.2520 - accuracy: 0.9228 - val_loss: 1.3639 - val_accurac
Epoch 4/50
104/104 [==============================] - 30s 290ms/step - loss: 0.1799 - accuracy: 0.9430 - val_loss: 0.7519 - val_accurac
Epoch 5/50
104/104 [==============================] - 30s 291ms/step - loss: 0.1457 - accuracy: 0.9536 - val_loss: 0.7128 - val_accurac
Epoch 6/50
104/104 [==============================] - 30s 290ms/step - loss: 0.1210 - accuracy: 0.9601 - val_loss: 0.7814 - val_accurac
Epoch 7/50
104/104 [==============================] - 30s 290ms/step - loss: 0.0995 - accuracy: 0.9689 - val_loss: 0.8483 - val_accurac
Epoch 8/50
104/104 [==============================] - 30s 290ms/step - loss: 0.0921 - accuracy: 0.9690 - val_loss: 0.8672 - val_accurac
Epoch 9/50
104/104 [==============================] - 30s 291ms/step - loss: 0.0764 - accuracy: 0.9740 - val_loss: 0.9111 - val_accurac
Epoch 10/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0698 - accuracy: 0.9780 - val_loss: 1.0067 - val_accurac
Epoch 11/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0693 - accuracy: 0.9782 - val_loss: 0.9390 - val_accurac
Epoch 12/50
104/104 [==============================] - 30s 291ms/step - loss: 0.0530 - accuracy: 0.9818 - val_loss: 1.0064 - val_accurac
Epoch 13/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0533 - accuracy: 0.9832 - val_loss: 1.0067 - val_accurac
Epoch 14/50
104/104 [==============================] - 30s 291ms/step - loss: 0.0489 - accuracy: 0.9846 - val_loss: 0.9397 - val_accurac
Epoch 15/50
104/104 [==============================] - 30s 291ms/step - loss: 0.0416 - accuracy: 0.9857 - val_loss: 1.0127 - val_accurac
Epoch 16/50
104/104 [==============================] - 32s 305ms/step - loss: 0.0412 - accuracy: 0.9853 - val_loss: 1.0367 - val_accurac
Epoch 17/50
104/104 [==============================] - 32s 304ms/step - loss: 0.0444 - accuracy: 0.9861 - val_loss: 1.0622 - val_accurac
Epoch 18/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0384 - accuracy: 0.9862 - val_loss: 1.1121 - val_accurac
```

```
Epoch 19/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0438 - accuracy: 0.9855 - val_loss: 1.0824 - val_accurac
Epoch 20/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0368 - accuracy: 0.9873 - val_loss: 1.0998 - val_accurac
Epoch 21/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0390 - accuracy: 0.9871 - val_loss: 1.1374 - val_accurac
Epoch 22/50
104/104 [==============================] - 30s 291ms/step - loss: 0.0398 - accuracy: 0.9863 - val_loss: 1.0858 - val_accurac
Epoch 23/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0390 - accuracy: 0.9868 - val_loss: 1.1076 - val_accurac
Epoch 24/50
104/104 [==============================] - 31s 295ms/step - loss: 0.0336 - accuracy: 0.9872 - val_loss: 1.1432 - val_accurac
Epoch 25/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0362 - accuracy: 0.9874 - val_loss: 1.1593 - val_accurac
Epoch 26/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0391 - accuracy: 0.9867 - val_loss: 1.1350 - val_accurac
Epoch 27/50
104/104 [==============================] - 30s 291ms/step - loss: 0.0358 - accuracy: 0.9870 - val_loss: 1.0937 - val_accurac
Epoch 28/50
104/104 [==============================] - 30s 292ms/step - loss: 0.0360 - accuracy: 0.9886 - val_loss: 1.1211 - val_accurac
Epoch 29/50
```

```
1 model_deep.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 1024)              22066176
_________________________________________________________________
dropout (Dropout)            (None, 1024)              0
_________________________________________________________________
batch_normalization (BatchNo (None, 1024)              4096
_________________________________________________________________
dense_1 (Dense)              (None, 512)               524800
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               131328
```

```
dropout_2 (Dropout)          (None, 256)               0
_________________________________________________________________
batch_normalization_1 (Batch (None, 256)               1024
_________________________________________________________________
dense_3 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_3 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 29)                3741
=================================================================
Total params: 22,764,061
Trainable params: 22,761,501
Non-trainable params: 2,560
_________________________________________________________________
```

```
print("Accuracy Score = ", model_deep.evaluate(X_test, to_categorical(label_encoder.transform(y_test))))
```

```
180/180 [==============================] - 7s 39ms/step - loss: 1.0485 - accuracy: 0.8445
Accuracy Score =  [1.0485262870788574, 0.8445217609405518]
```

```
predictions = list(model_deep.predict_classes(X_))
deep_tags = []
for i in predictions:
  deep_tags.extend(list(label_encoder.inverse_transform([i])))
```

```
WARNING:tensorflow:From <ipython-input-21-d8ceb1f49df6>:1: Sequential.predict_classes (from tensorflow.python.keras.engine.sequ
Instructions for updating:
Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g. if it uses
```

```
with open ('/content/drive/MyDrive/Cleaned_Sentences_Task/ann_predictions.txt', 'w+') as file:
    json.dump(deep_tags, file)
```