Team Project

Group ID: 64, Paper ID: 79

[Code]

[Final Report]

Zerui Tian - zeruit2@illinois.edu

Yixing Zheng - yixingz3@illinois.edu

Intro to Paper:
Disease Inference with Symptom Extraction and Bidirectional Recurrent Neural Network

- This paper aims to provide a new effective disease inference method utilizing symptoms extracted from Electronic Medical Records (EMR) data.
- The relationship between symptoms and diseases is represented by the term frequency-inverse document frequency (TF-IDF) model. And a bidirectional recurrent neural network (Bi-LSTM) is utilized to model the symptom sequences in EMR data.
- This combination of models shows a significant improvement in disease inference - 4\% to 10\% on average improvement from the two baseline models (in the paper, DeepLabeler and WordVec + Bi-LSTM).

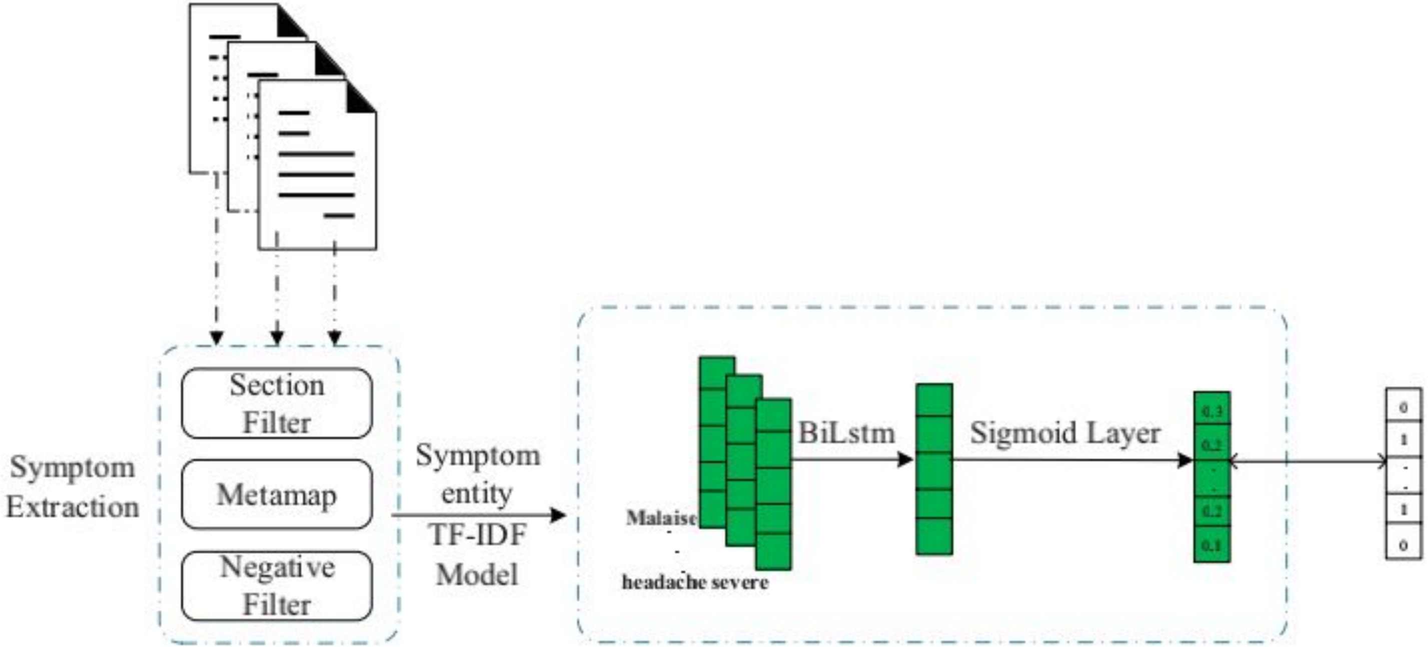


Fig.1. Overview of disease inference process.

Methodology

Data:

- The IMDB data set

Models:

- Proposed
    - TF-IDF + Bi-LSTM
- Baseline
    - WordVec + Bi-LSTM
    - Replaced DeepLabeler with
        - Glove + Bi-LSTM
    - Introduced additional model:
        - Regular Embedding Layer

Embedding methodology

- TF-IDF: $W_{i,j} = TF_{i,j} * \log \frac{N}{D_i}$

- Word2Vec: It is used to find embedding vectors such that similar words are close together and dissimilar words are far apart.

- GloVe: It essentially reduces the dimensionality of the co-occurrence matrix and learns the low-dimensional representation of the word.

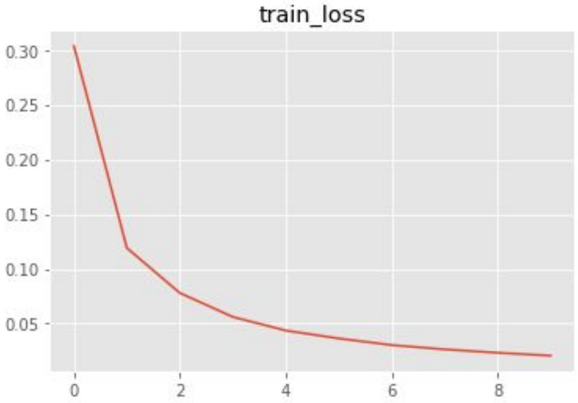
Network Structure

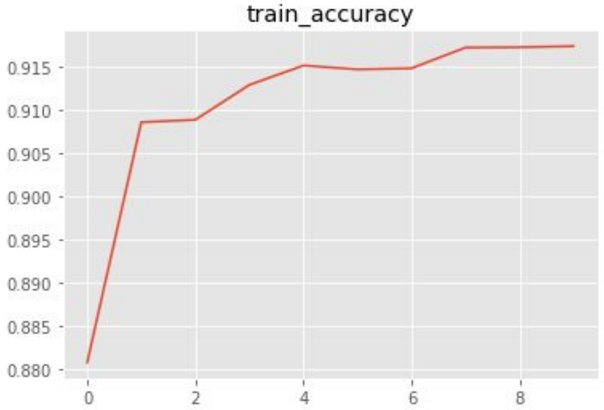
Bi-LSTM

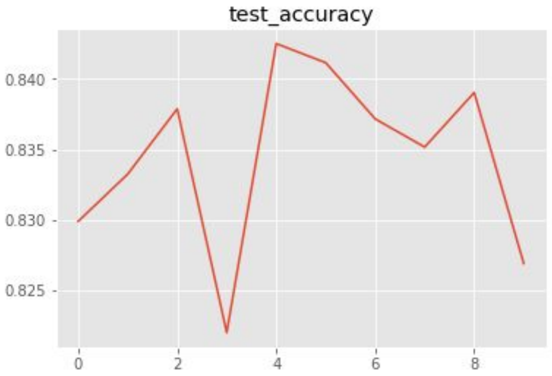
```
# Create Bi-LSTM model

class BiLSTM(nn.Module):
  def __init__(self, embed_size, num_hiddens, num_layers):
    super(BiLSTM, self).__init__()
    self.embedding = nn.Embedding(len(vocab),embed_size)
    self.LSTM = nn.LSTM(input_size=embed_size,hidden_size=num_hiddens,num_layers=num_layers,bidirectional=True)
    self.fc = nn.Linear(4*num_hiddens, 2)
    self.dropout = nn.Dropout(0.8)
  def forward(self, inputs):
    #inputs = inputs.float()
    embeddings = self.embedding(inputs.permute(1,0))
    outputs, _ = self.LSTM(embeddings)
    outputs = self.dropout(outputs)
    result = torch.cat((outputs[0], outputs[-1]), -1)
    result = self.fc(result)
    return result
```

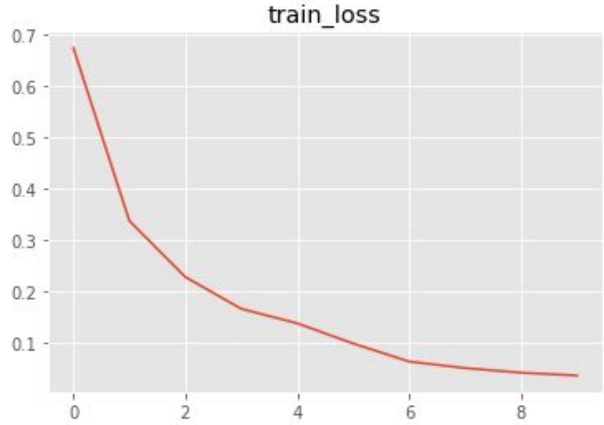
Results for TF-IDF

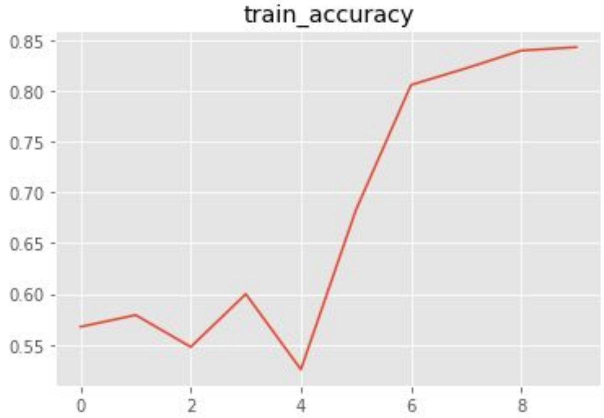


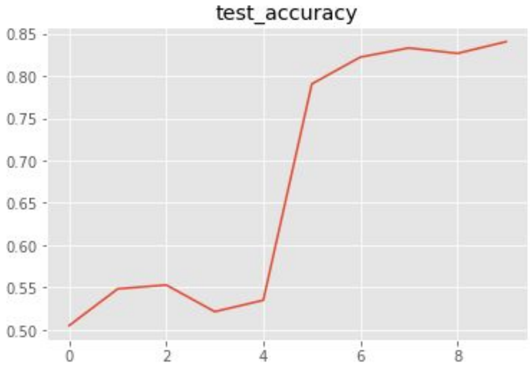




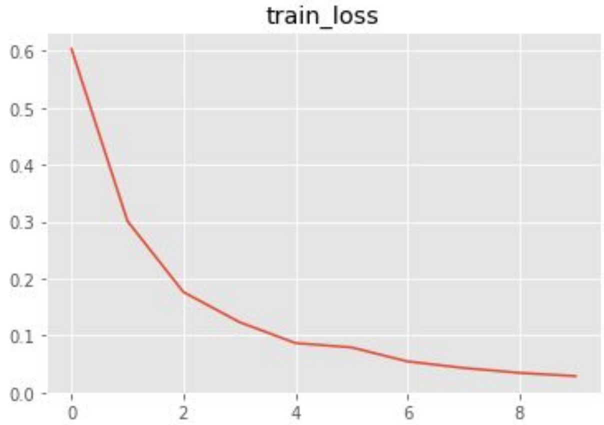
Results for GloVe

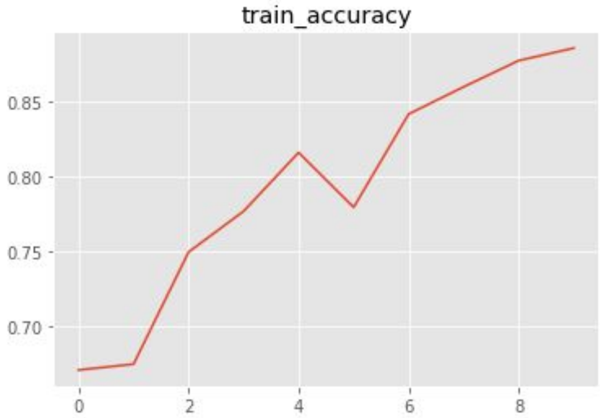


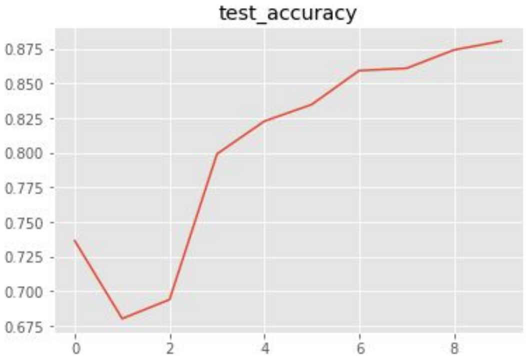




Results for Word2Vec







Results - Mean

Claim verified: TF-IDF + Bi-LSTM performs better than the baseline models with ~4% - 10% improvement.

Model	Loss	Train Acc	Test Acc	Runtime
TF-IDF + Bi-LSTM	**0.0738**	**0.9108**	**0.8345**	~10 Mins
Glove + Bi-LSTM	0.1836	0.6814	0.6776	~10 Mins
WordVec + Bi-LSTM	0.1528	0.7929	0.8042	~10 Mins
Embedding	0.1429	0.8331	0.7682	~10 Mins

Results - Last Epoch

Model	Loss	Train Acc	Test Acc	Runtime
TF-IDF + Bi-LSTM	0.0208	0.917	0.827	~1 Min
Glove + Bi-LSTM	0.0363	0.843	0.840	~1 Min
WordVec + Bi-LSTM	0.0285	0.885	**0.881**	~1 Min
Embedding	**0.0128**	**0.955**	0.811	~1 Min