Task report

Oleksandr Marmaliuk

1

oleksandrmarmaliuk@gmail.com

1. Name Entity Recognition

Here we discuss potencial improvements of the work.

1.1. Fine-tuning the model

The LSTM-CRF model combines a sequence modeling component (LSTM) with a Conditional Random Field layer. Fine-tuning the model can involve strategies such as adjusting hyperparameters, modifying the architecture, and optimizing the training process.

Firstly, adjusting hyperparameters is a crucial stem for improving the model's performance:

- Learning Rate: Experiment with different learning rates to balance convergence speed and stability. Learning rate schedulers, such as ReduceLROnPlateau, can dynamically adjust the learning rate.
- Dropout Rate: Fine-tune the dropout rate to prevent overfitting.
- Hidden Dimensions: Increasing the LSTM's hidden dimensions can enhance its capacity to learn complex patterns but may also lead to overfitting.
- Batch Size: Experiment with batch sizes to balance gradient estimates and training stability.

Notably, we can use grid search, random search or Bayesian optimization for adjusting hyperparameters optimally.

Secondly, we can adjust the model architecture:

- Pretrained embeddings usage: replace randomly initialized embeddings with pretrained embeddings, such as GloVe, BERT or FastText, for improved initialization.
- Attention Mechanism: Add an attention layer after the LSTM to dynamically weigh important tokens.
- Dropout in LSTM Layers: Include dropout within LSTM layers to reduce overfitting.

Wide range of measures could be taken to make model work with a better prepared data.

- Finding optimal padding, encoding, tokenization.
- Class Balancing: Tackling class imbalance problem.
- **Class set:** possibly, looking for a dataset with wider variety of plasses annotated.

We should also find the best optimizer and fine-tune optimizer parameters.

We may also try Transfer learning approach: fine-tune a pretrained LSTM-CRF model on the target dataset.

1.2. Building HMM

Another approach, which I unfortunately did not implement is building a Hidden Markov Model. However, HMMs are not compatible with the dataset I started working with due to an extremely low number of represented classes.

HMMs are effective with seqential data, where we are willing to detect certain patterns in the ordering of the objects within sequences.