Deep Neural SCOTUS: A Comparative Analysis of Language-Modeling Techniques for US Supreme Court Opinions

Andrew Miller-Smith, MS Statistics
aams@stanford.edu

Stanford
Class: CS 224N

Background and Overview

Language modeling plays a key role in computational linguistics, featuring prominently in many NLP applications. Recent work has focused on sub-word constructs, incorporating them into models with impressive results. In this study I compare two models, a word-based architecture and a similar one enhanced with character encodings.

Data

Kaggle hosts a data set of approximately 35,000 United States Supreme Court (SCOTUS) rulings from 1789 to 2017.

- Files include majority, dissenting, etc. opinions from 96 justices
- Modern opinions frequently employ archaic diction and formatting
- Older writings burst with anachronisms and misspellings
- Train/dev/test sizes: 5M/40K/40K

Sample sentence and inputs

"All the California Supreme Court's decision stands for is that, so far as California is concerned, petitioners may ``assert legal arguments in defense of the state's interest in the validity of the initiative measure'' in federal court. 628 F. 3d 1191, 1193."

Inputs	Target
All the California Supreme Court	'
the California Supreme Court '	s
California Supreme Court ' s	decision
Supreme Court ' s decision	stands

Models

Both models used pre-trained GloVe vectors for words, and the combined one learned its own character embeddings.

Word model

- Inputs move through a bi-LSTM
- Forward and backward hidden states concatenated to preserve information
- Result sent through a dropout layer and an activation-less highway layer
- Projected into vocabulary space and evaluated via cross-entropy loss

Combined model

- Character representations pass through a CNN followed by LeakyRELU, activation, max pooling, and dropout
- Word representations pass through word model (bi-LSTM -> dropout -> highway), are then projected into a single encoding
- Representations concatenated, projected to size of word embedding
- Travel through activation-less highway layer then mapped to vocabulary space where evaluated via same metric

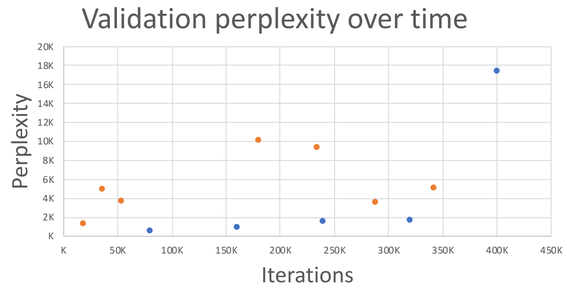
Hyperparameters

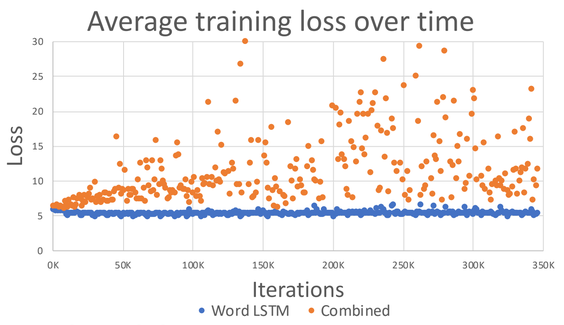
Factors tuned while training

- Window size: 5
- Learning rate: 0.01
- Dropout probability: 0.3
- Word embedding size: 200
- Character embedding size: 256

Results

The combined model failed to match even the worst performance of the word LSTM, though both struggled overall.





Word model

- Model failed to improve on data sets larger than 500K inputs
- Maintained constant training loss yet increasing validation perplexity
- Test perplexity: 258.60

Combined model

- Performance unstable without removal of highway or linear consolidation of bi-LSTM outputs
- Required ten times as many epochs to match word model
- Test perplexity: 2468.71

Takeaways

Key findings

- Word model significantly outperformed combined model
- Character-level information appears to have clashed with word information
- Neither model was able to perform well on validation set or test set

Error analysis

- Poor overall performance raises questions about data and model design/implementation
- Data might not follow standard IID assumptions
- Extensive investigation revealed no perceptible errors in model code
- Unclear why additional layers hurt combined model's performance
- Additional layers might have muddled signal from LSTM

Next steps

Analyzing the data set presented various challenges and additional avenues to explore.

Further exploration

- Better understand difficulty learning on data with more than 500K examples
- Explore a character-exclusive model, compare against the architectures tested in this experiment