Classifying Amazon Product Category Based on Description

Erdenebold Battulga, Kevin Darmawangsa

Overview

Motivation
Automatic Text Classification can be useful in various ML tasks such as text mining, content management, product review analysis, spam filtering, etc.
Problem
We're classifying items on Amazon based on their descriptions.
Data
We're using Amazon Review Data, provided by Julian McAuley from UCSD (He & McAuley, 2016).
Input:
Amazon product description
Output
Amazon product category
Approach
We decided to tackle this problem with various types of neural networks: simple Neural Network (NN), Convolutional Neural Network(CNN), Recurrent Neural Network(RNN), and RNN with Attention.

Methods

Feature Extraction
Depending on the Neural Network, we used various types of feature extraction methods.
Simple NN:
- Bag of Words (BoW)+term frequency inverse document frequency (TFIDF)
- Pre-trained 100D GloVe Embedding
- Model-trained 100D Embedding

CNN, RNN, RNN with Attention:
- Pre-trained 100D GloVe Embedding
- Model-trained 100D Embedding

Network Design
We expressed network design as hyperparameters and used grid search to find the optimal design.
Simple NN: The design hyperparameters are number of dense layers and the number of nodes in each layer.
CNN: The design hyperparameters are the number of filters and kernel size for each convolutional layer.
RNN: The design hyperparameters are the number of LSTM cells and number of dense layers.
Activation Function: We used softmax for the final output layer and standard ReLU for the other layers.

Discussion

- For simple NN and RNN, increasing input length increases accuracy but is capped at 1000. CNN works well with all input lengths.
- Across all neural networks, adding pre-trained GloVe Embedding increased accuracy. However, we found that models with Embedding learned during training perform even better.
- In simple NN, increasing the number of hidden layers decreased accuracy, hence shallow network fitted better with the data.
- In CNN, higher kernel size and filter number give higher accuracy.
- In RNN, bidirectional performs marginally better than unidirectional.
- In RNN, number of LSTM hidden units positively correlated with accuracy but plateaued around 128 cells.
- Many hyperparameters, including number of fully connected hidden layers, pooling layers, dropout layer, batch size, optimizer type did not influence test accuracy significantly.

NN	CNN	RNN
6.3x	1.0x	17.0x

Table 1. Comparison of Average Runtime for the Best Models

- With the same feature extraction, CNN ran the fastest.

NN	CNN	RNN	RNN+Attention
74.8	77.8	78.9	79.4

Table 2. Comparison of Best Test Accuracy

- Overall, given the entire training dataset, RNN performed slightly better than CNN, but at the cost of longer training period.
- We also tried adding attention layer to the RNN model and accuracy was slightly improved.
- One challenge of the project was figuring out all of the different hyperparameters and systematically varying and keeping track of all the tuned models.

Results

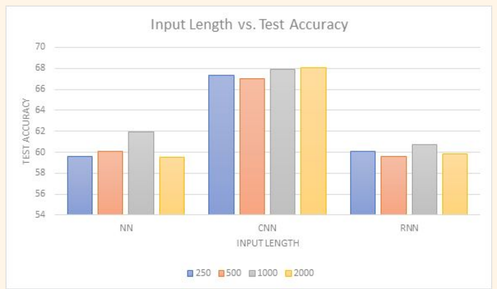

Figure 1. Comparison of Different Input Lengths

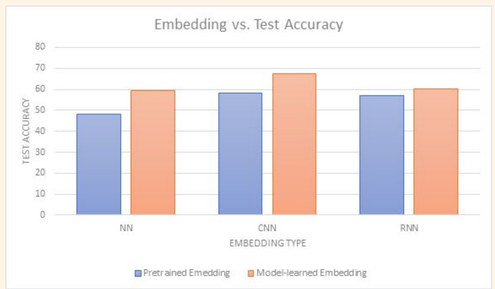

Figure 2. Comparison of Different Embeddings

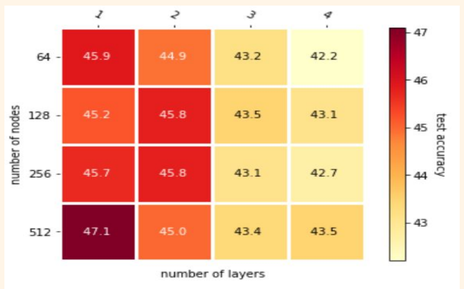

Figure 3. Varying number of nodes and layers in NN

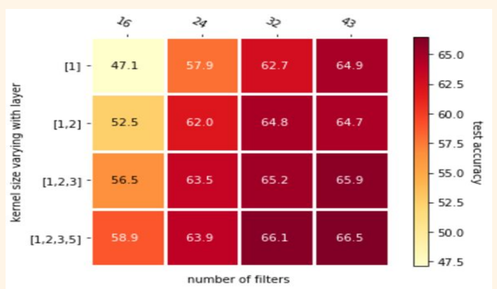

Figure 4. Varying kernel size and number of filters in CNN

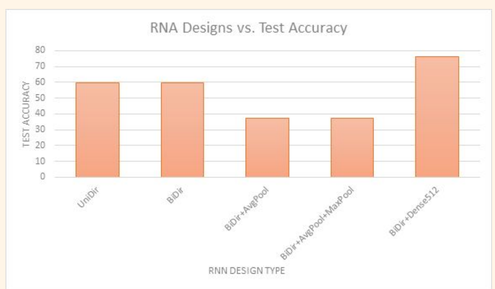

Figure 2. Comparison Various RNN Designs

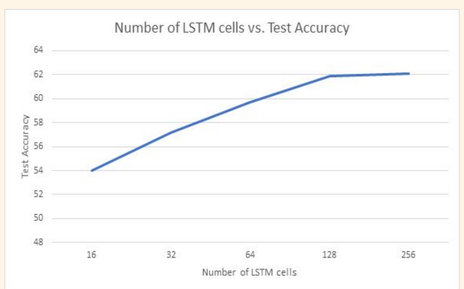

Figure 6. Varying number of LSTM cells in RNN

References

He, R., & McAuley, J. (2016). Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. *Proceedings of the 25th International Conference on World Wide Web - WWW '16*, 507–517. https://doi.org/10.1145/2872427.2883037

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *ArXiv:1408.5882 [Cs]*. Retrieved from http://arxiv.org/abs/1408.5882

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical Attention Networks for Document Classificatiaon. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480–1489. https://doi.org/10.18653/v1/N16-1174