Comparative Evaluation of Pretrained Transfer Learning Models on Automatic Short Answer Grading

June 22nd, 2020

Authors
Sasi Kiran Gaddipati
Deebul Nair
Paul G. Plöger

Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Introduction

Automatic Short Answer Grading (ASAG)

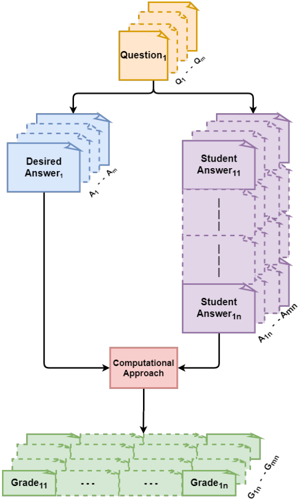

Figure 1: Automatic short answer grading procedure

Word Representations

Word Embeddings

Also known as "word vectors" and "word representations"

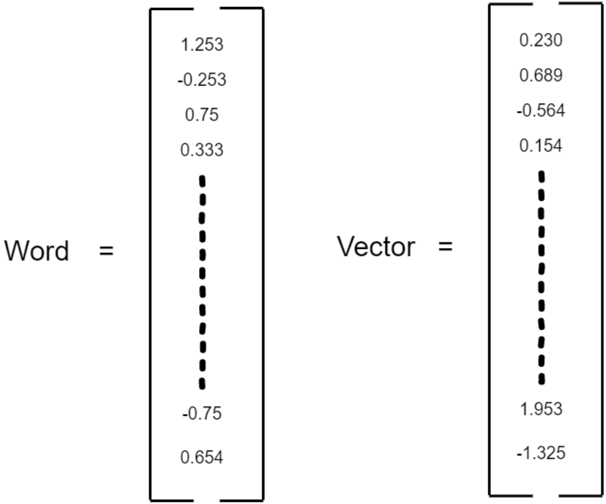

Figure 2: Representation of words as high-dimensional vectors

Conventional Methods

Word2Vec[1]

- Distributional representation of words
- Cannot represent a word out of training vocabulary
- Conditional probability approach

GloVe[2]

- Co-occurrence matrix of words
- Documents $\rightarrow$ Co-occurrence matrix $\rightarrow$ Word embeddings
- Cannot represent a word out of training vocabulary

FastText[3]

- Pretraining on character n-grams
- Can create vectors for the words out of training vocabulary

Drawbacks

- Ignore the word's context
- Training from scratch for every task
- Ineffective with long-term dependencies
- Can not assign word vectors for unknown words

Transfer Learning

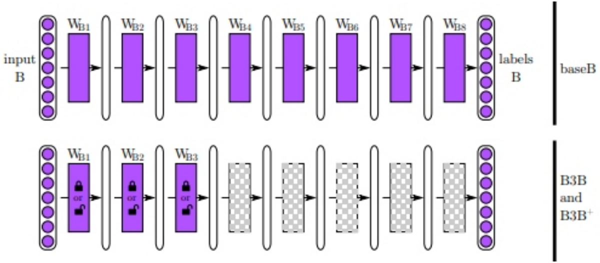

Figure 3: Transfer learning[4]

- Reduces computing cost and time
- Better results with limited task data
- No training from scratch every time

Transfer Learning in Natural Language Processing

Pretrained Transfer Learning Models

Embeddings from Language Models (ELMo)[5]

- **Architecture:** Three layers of Long Short-Term Memory (LSTM)[6]
- Extracts both syntactic and semantic features
- Allot different word vectors for homonyms in different contexts

Generative Pretraining (GPT)[7]

- **Architecture:** Stacked transformer architecture[8]
- Architecture provides structured memory for long-term dependencies[7]
- Semi-supervised approach

Pretrained Transfer Learning Models II

Bidirection Encoder Representations from Transformers (BERT)[9]

- **Architecture:** Stacked bi-directional transformer architecture
- Pretraining with:
  - Masked Language Model (MLM)
  - Next sentence prediction

GPT-2[10]

- **Architecture:** Stacked transfromer architecture
- Extension of GPT architecture
- Unsupervised pretraining on language modeling

Overview of Transfer Learning Models

Table 1: An overview of transfer learning models

Model	Architecture	Dataset name	Dataset size
ELMo[5]	bi-LSTM	One billion word benchmark[11]	1B words
GPT[7]	Transformer	BookCorpus[12]	800M words
BERT[9]	Transformer	BookCorpus[12] English Wikipedia	800M words 2500M words
GPT2[10]	Transformer	WebText[10]	-

Dataset

Mohler Dataset

- Domain-specific data with computer science questions and answers
- Evaluates the students' answers by comparing them with the desired answer

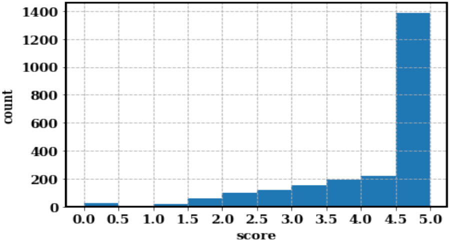

Figure 4: Histogram representation of the count of grades in Mohler dataset

- Graded each student answer from 0 (not correct) to 5 (totally correct)
- Biased dataset with mean (4.16) and median (4.50) of average assigned grades

Experimentation

Preprocessing

- Applied **word tokenization**

```
sentence = 'This is an example tokenized sentence.'
print(word_tokenize(sentence))
```

```
['This', 'is', 'an', 'example', 'tokenized', 'sentence', '.']
```

Figure 5: An example of tokenizing a sentence

- No **spell checking** implemented
- To analyze the performance of pretrained embeddings in raw setup:
  - **Lemmatization** and **stop word removal** are neglected
  - **Question demoting** is avoided, in contrast to former works[13][14][15]

Feature Extraction

Sum of Word Embeddings (SOWE) for an answer j of question q_i is given by

$$\boxed{a_{ij} = \sum_{j=1}^{n_j} w_k} \tag{1}$$

where a_{ij} represents the j^{th} answer vector of question q_i and n_j represents the number of words in the answer

A single feature, cosine similarity, is computed between each student answer vector a_{ij} and desired answer vector a_i

$$\boxed{cos(a_{ij}, a_i) = \frac{a_{ij}.a_i}{|a_{ij}||a_i|}} \tag{2}$$

Training

- Split the Mohler data 70% for training and 30% for testing randomly
- Train the data on three regression models: isotonic, linear and ridge (non-linear)
- Implement the selected regression models to compare our results with former works[13][15][16]
- Train the cosine similarity feature with the correspondingly assigned grades

Testing

- Test data is unseen by the regression model until it's testing phase
- Similarity scores of test data are input through the trained regression model
- Results in the predicted grades will be further used for evaluation
- Repeat the experiment over 1000 iterations
- Compute the Root Mean Square Error (RMSE) and Pearson correlation (ρ)

Metrics

Root Mean Square Error (RMSE)

$$\boxed{RMSE(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}} \tag{3}$$

- Absolute error between the actual and predicted value
- RMSE is proportional to the square of the error
- Lower the RMSE, better the model
- Suboptimal metric for semantic text similarity

Pearson Correlation

$$\boxed{\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}} \tag{4}$$

where *cov* and σ represents the covariance and standard deviation respectively

- Correlation measures the subjective and relative nature of the semantic scores
- Defines the linear correlation between the two random variables
- Higher the correlation, better the model

Results

Pearson Correlation

Table 2: Pearson correlation results of transfer learning models on Mohler dataset

Model	Isotonic regression	Linear regression	Ridge regression
ELMo	**0.485**	**0.451**	**0.449**
GPT	0.248	0.222	0.217
BERT	0.318	0.266	0.269
GPT-2	0.311	0.274	0.269

Root Mean Square Error

Table 3: RMSE results of transfer learning models on Mohler dataset

Model	Isotonic regression	Linear regression	Ridge regression
ELMo	**0.978**	**0.995**	**0.996**
GPT	1.082	1.088	1.089
BERT	1.057	1.077	1.075
GPT-2	1.065	1.078	1.079

Overview of Results I

The features in Table 4 are represented with acronyms and colors for better distinction as follows:

Acronym	Feature
SVMRank	Support Vector Machine Rank
SVR	Support Vector Regression
LR	Length Ratio of student answer to desired answer
SIM	Similarity between student answer and desired answer
SOWE	Sum of word embeddings of answers
VP	Verb phrases

Overview of Results II

Table 4: Overview comparison of results on Mohler dataset

Model/Approach	Features	RMSE	Pearson correlation
BOW[13]	SVMRank	1.042	0.480
	SVR	0.999	0.431
tf-idf[13]	SVMRank	1.022	0.327
	SVR	1.022	0.327
tf-idf[14]	LR + SIM	**0.887**	**0.592**
Word2Vec[15]	SOWE + VP	1.025	0.458
	SIM + VP	1.016	0.488
GloVe[15]	SOWE + VP	1.036	0.425
	SIM + VP	1.002	0.509
FastText[15]	SOWE + VP	1.023	0.465
	SIM + VP	0.956	0.537
ELMo	SIM	0.978	0.485
GPT	SIM	1.082	0.248
BERT	SIM	1.057	0.318
GPT-2	SIM	1.065	0.311

Conclusion

Observations

- Results illustrate the robustness of the ELMo model on the domain-specific dataset
- BERT and GPT performed below par in all the cases
- Reasons that ELMo may have worked better is two fold:
  - Assignment of different vectors for the same word in different contexts
  - Significant presence of domain data in the pretrained corpus
- Linear and ridge regression results of transfer learning models' are similar:
  - Due to the significant linear fit of the data
  - Reduced non-linear features between the training variables

Future Work

- Pretrain or finetune with domain-specific data
- Implementation of further preprocessing such as
  - Question demoting
  - Stop word removal
- Intense feature extraction
  - Similar word alignment[16]
  - Length ratios
- Usage of universal sentence encoders[17]
- Alternate dataset usage to Mohler (highly biased)

Questions??

References I

[1] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: arXiv preprint arXiv:1301.3781 (2013).

[2] Jeffrey Pennington, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation". In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014, pp. 1532–1543.

[3] Piotr Bojanowski et al. "Enriching word vectors with subword information". In: Transactions of the Association for Computational Linguistics 5 (2017), pp. 135–146.

[4] Jason Yosinski et al. "How transferable are features in deep neural networks?" In: Advances in neural information processing systems. 2014, pp. 3320–3328.

[5] Matthew E Peters et al. "Deep contextualized word representations". In: arXiv preprint arXiv:1802.05365 (2018).

[6] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: Neural computation 9.8 (1997), pp. 1735–1780.

[7] Alec Radford et al. "Improving language understanding by generative pre-training". In: URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf (2018).

[8] Ashish Vaswani et al. "Attention is all you need". In: Advances in neural information processing systems. 2017, pp. 5998–6008.

[9] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: arXiv preprint arXiv:1810.04805 (2018).

[10] Alec Radford et al. "Language models are unsupervised multitask learners". In: OpenAI Blog 1.8 (2019).

[11] Ciprian Chelba et al. "One billion word benchmark for measuring progress in statistical language modeling". In: arXiv preprint arXiv:1312.3005 (2013).

[12] Yukun Zhu et al. "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books". In: Proceedings of the IEEE international conference on computer vision. 2015, pp. 19–27.

References II

[13] Michael Mohler, Razvan Bunescu, and Rada Mihalcea. "Learning to grade short answer questions using semantic similarity measures and dependency graph alignments". In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics. 2011, pp. 752–762.

[14] Md Arafat Sultan, Cristobal Salazar, and Tamara Sumner. "Fast and easy short answer grading with high accuracy". In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016, pp. 1070–1075.

[15] Tim Daniel Metzler. "Computer-assisted grading of short answers using word embeddings and keyphrase extraction". In: Hochschule Bonn Rhein Sieg Master Thesis (2019).

[16] Ramesh Kumar. Evaluation of Semantic Textual Similarity Approaches for Automatic Short Answer Grading. 2018.

[17] Daniel Cer et al. "Universal sentence encoder". In: arXiv preprint arXiv:1803.11175 (2018).