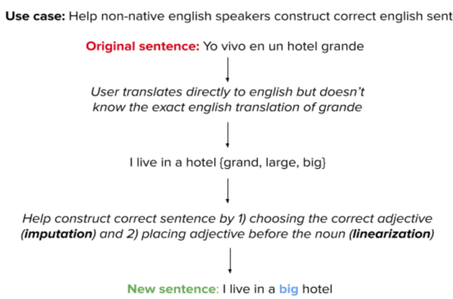
TEXTIFY: A WORD IMPUTATION AND LINEARIZATION TOOL

Avanika Narayan, Anmol Tukrel, Joshua Lara
Stanford University - Department of Computer Science
{avanikan,anmol.tukrel,joshlara}@stanford.edu

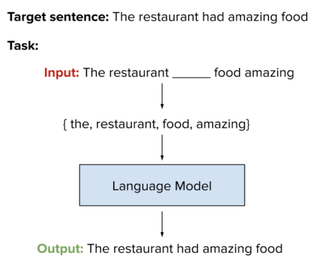
Motivation

Non-native English speakers have trouble filling out online forms, writing documents, or communicating online in English. Oftentimes, these speakers misorder words or fail to add critical parts of speech when constructing sentences. Techniques in natural language processing show promise in helping build tools for grammar-correction that could provide support to such individuals.



Task

The goal of this project is to construct a tool that is capable of automatically producing grammatically correct sentences by jointly addressing the tasks of **word linearization** and **word imputation**.



Data and Features

Data Source: MSR/Holmes Dataset
Details:

- Training dataset: 500 full-length 19th century books from the Gutenberg Project.
- Testing dataset: 1,040 sentences from Sherlock Holmes novels.
  - Structure of test data: "I have seen it on him , and could _____ to it." [write, migrate, climb, swear, contribute]

Data Source: Yelp Reviews
Details:

- Training dataset: 1,561,488 sentences from Yelp reviews
- Testing dataset: 195,185 sentences

Methods

Approach 1: Language Model Beam-Search with Future Cost Heuristic

- Scoring Function

$$f(x,y) = \sum_{n=1}^{N} \log p(x_{y(n)} \mid x_{y(1)}, \ldots, x_{y(n-1)})$$
$$y^* = \arg\max_{y \in \mathcal{Y}} f(x,y)$$

Fig. 3: Scoring function

- Language Models:
  - N-grams, Pre-trained awd-lstm-lm, Yelp-LSTM (Yelp dataset specific language model), MSR-LSTM (MSR dataset specific language)

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 207, 20)	3435700
dropout_1 (Dropout)	(None, 207, 20)	0
lstm_1 (LSTM)	(None, 256)	283648
dense_1 (Dense)	(None, 171784)	44148488

Fig. 4: Yelp-LSTM network

- Future cost heuristic: Unigram cost of uncovered tokens

Approach 2: Natural Language Generation Denoising Auto-Encoder (NLG-DAE)

- Key idea: introduce noise into training examples (remove + re-order words) such that the resulting model can generate correct sentence reconstructions from noisy inputs
- Network Architecture:
  - Encoder - Bidirectional RNN
  - Decoder - Attention-based RNN
    - attention weights computed via 2-layer feed-forward neural network

Implementation + Experiments

Language Model Training

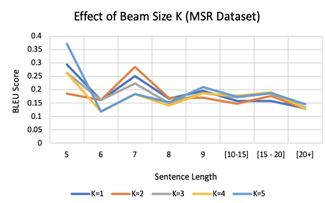
- Yelp-LSTM: Trained on 20K sentences (139,595 sequences) for 50 epochs
- MSR-LSTM: Trained on 10K sentences (230,931 sequences) for 25 epochs
- NLG-DAE: Trained on 20K sentences for 5 epochs
  - Training data generation: 1) remove one random word and 2) shuffle words
- N-grams: Trained on 1,048,576 sentences

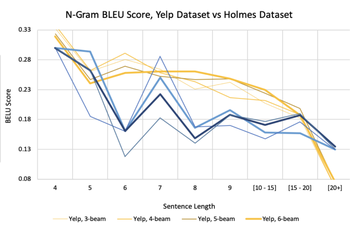
Experiments:

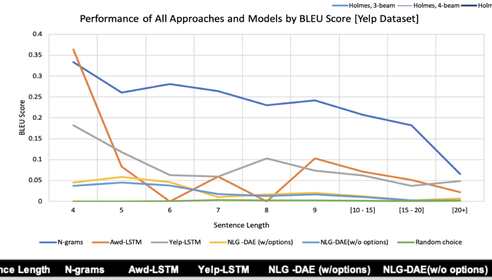
- **Approach 1**: Language Model Beam-Search
  - Varied beam size: 3 - 5
  - 3 different language models: N-grams, awd-lstm-lm, Yelp-LSTM, MSR-LSTM
- **Approach 2**: NLG-DAE:
  - Options for missing word provided and output constrained to words only in the original sentence
  - No missing word options provided and no constraints enforced on output

Results

BLEU Score: score based on n-gram matches between original and constructed sentence







Sentence Length	N-grams	Awd-LSTM	Yelp-LSTM	NLG -DAE (w/options)	NLG-DAE(w/o options)	Random choice
4	0.3333	0.3636	0.1818	0.045	3.67E-02	2.32E-156
5	0.2607	0.0833	0.1179	0.0588	4.54E-02	5.54E-80
6	0.2807	0	0.0625	0.0461	3.78E-02	9.90E-04
7	0.2638	0.0591	0.0591	0.01018	0.0174	0.003
8	0.2306	0	0.1028	0.0164	0.0134	0.0024
9	0.242	0.1026	0.0733	0.0203	0.0168	0.0024
[10 - 15]	0.2069	0.0709	0.0619	0.01246	0.0102	0.0014
[15 - 20]	0.1826	0.0514	0.0372	0.0026	0.0019	0.0007
[20+]	0.0658	0.0215	0.0485	0.0065	0.0019	0.001

Conclusion + Future Work

- Language models that are trained on more data (even if they are less complex) achieve better result on the desired task
  - Yelp reviews are varied in language (word choice + prose) and domain (business type), so training on 20K examples was insufficient
  - Due to **constrained access to computing power**, we were unable to train on more examples for a longer duration of time
- Similarity between training dataset and test dataset matters
- Tuning beam size had negligible difference on accuracy for beam size > 3.
- **Future work**:
  - Train NLG-DAE and Yelp-LSTM on more data and for longer
  - Perform hyper-parameter tuning to find optimal learning rate, # of hidden units etc.
  - Perform inference on NLG-DAE w/beam-search

Acknowledgements

We would like to thank our mentor, Richard Diehl, for his support in this project!