Team 2

Haruna Tanaka, Nicholas Mehr, Toshinori Kitamura,
Xiaobai Sun, Masayuki Takeda, Kai Ru

Intro to NLP and Language Models

Part 1 NLP, Applications, Challenges

NLP

The process of creating algorithms that transform some language input like text or audio into words, labeling them based on the position and function of the words in the sentence.

Continuous Bag of Words model (CBOW)

Word embeddings

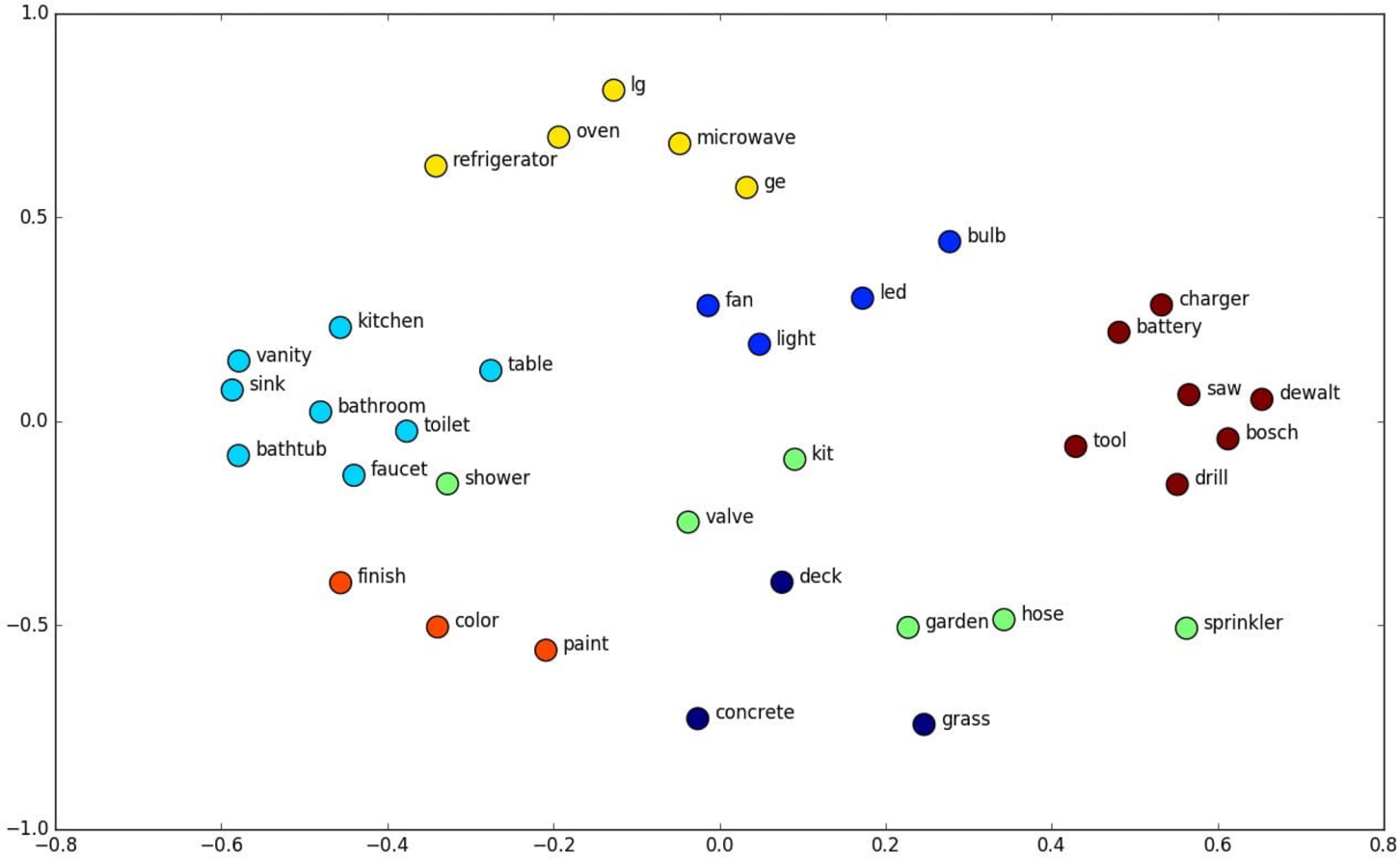
Continuous Bag of Words model (CBOW)

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

The resulting counts matrix will then be:

$$X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array} \begin{array}{c} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Word embeddings, an N-dimensional vector representation of human words, are needed to create meaningful inputs.



Applications

1. Machine Translation
2. Speech Recognition
3. Sentiment Analysis
4. Question-Answering
5. Automatic Summation
6. Chat-bots
7. Market Intelligence
8. Text Classification
9. Character Recognition
10. Spell Checking

Intro to NLP and Language Models

Part 2: Statistical vs neural networks and conventions of neural network architecture

1) **Statistical Language Models**

- N-grams
- Hidden Markov Models
- Hard-coded linguistic rules

Limitations of N-gram models:

- Simplest language model, cannot achieve fluency
- Higher N leads to computational overhead
- Crude representation of language built on the probability of words co-occurring, i.e., gives zero probability to all the words that are not present in the training corpus.

N-Gram Models:

$$\text{Unigram LM} : p(w_1^N) = \prod_{n=1}^{N} p(w_n)$$

$$\text{Bigram LM} : p(w_1^N) = \prod_{n=1}^{N} p(w_n|w_{n-1})$$

$$\text{Trigram LM} : p(w_1^N) = \prod_{n=1}^{N} p(w_n|w_{n-2}, w_{n-1})$$

Rudimentary Example:
P(w|h**)**

P(the**|**its water is so transparent that**)**

Intro to NLP and Language Models

Part 2: Neural networks and conventions of neural network architecture

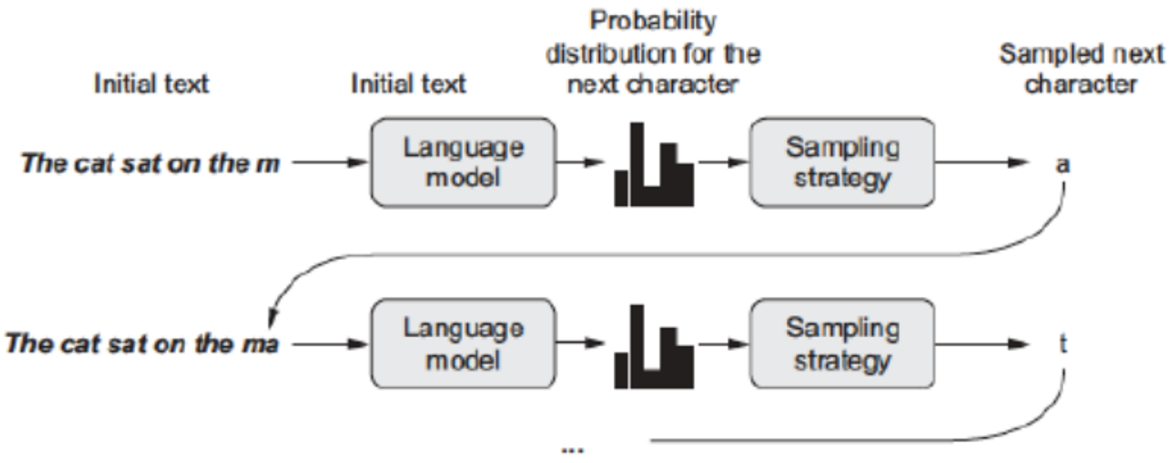
2) **Neural Language Models**

- LSTMs
- Bi-directional models/attention Mechanisms

Advantages over N-Gram models

- Can introduce randomness in the sampling process via stochastic sampling.
- Probabilistic sampling from Softmax output can generate new or realistic words not in training data

Implementing Character-Level Networks:

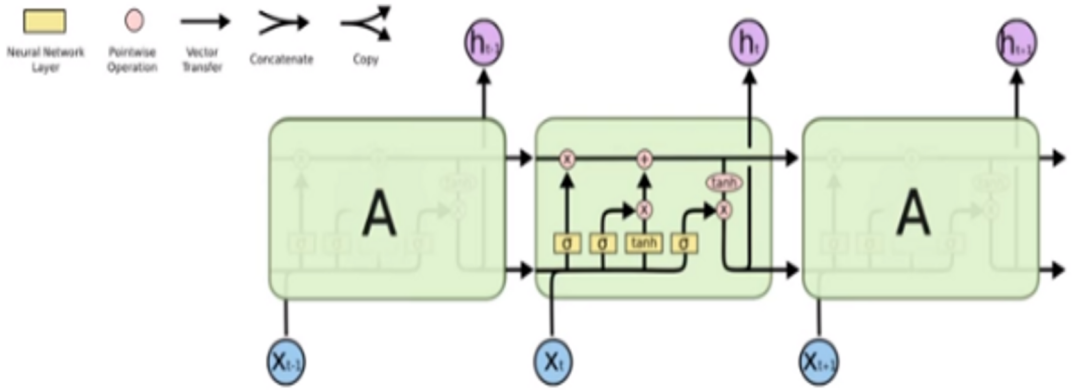


Intro to NLP and Language Models

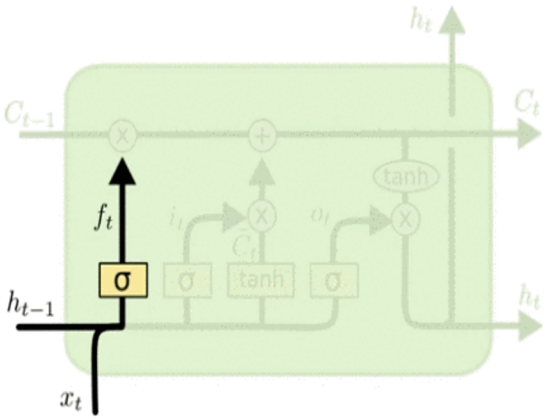
Part 2: Neural networks and conventions of neural network architecture

2) **Neural Language Models**

- LSTMs
  - Q&A Application (right)
  - Input/Forget/Output Gates (below)



Bob and Alice are having lunch. Bob likes apples. Alice likes oranges.
She is eating an orange.



(1) $f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$

Intro to NLP and Language Models

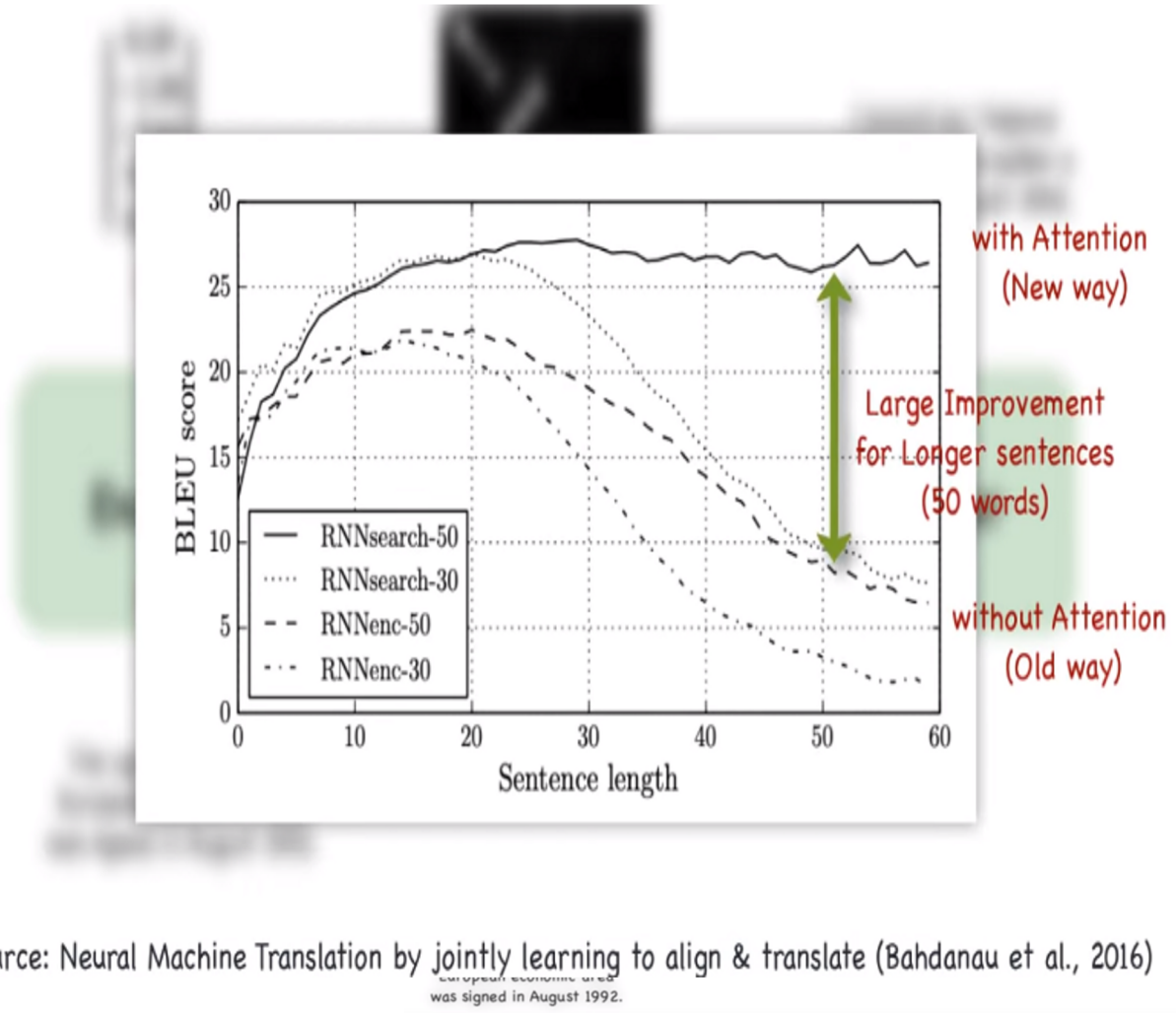
Part 2: Neural networks and conventions of neural network architecture

2) **Neural Language Models**

- Attention mechanisms

Why we need Attention mechanisms:

- Single-directional encoder-decoder does not adequately address complexity of grammar in longer sentences.
- Solution: make **bi-directional LSTM**
- New problem: **which word** do we need to focus on in a sequence?



Source: Neural Machine Translation by jointly learning to align & translate (Bahdanau et al., 2016)

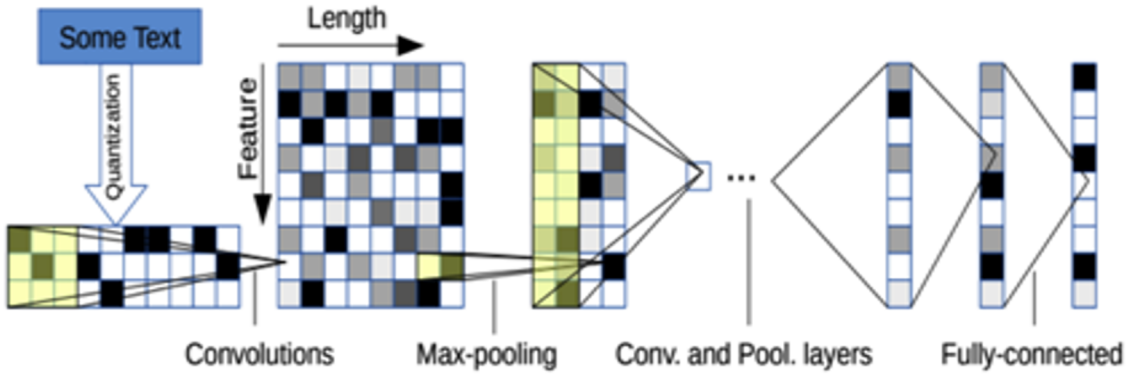
How Conventional Approach Deal With Language Model

CNN

- Convolution approaches to language models use probabilistic description of language phenomena.
- Probability comes in handy in determining
- Currently, nearly all techniques of text classification are word based.

Conventional Approach When It Is CNN

- Character-level Convolutional Networks for Text Classification(CNN) **treats characters on a sentence level**
- Need arises to prevent out of vocabulary problem (OOV)
- Differs from traditional approaches as works without any knowledge on the syntactic or semantic structures of a language.



Dealing With Language Model In CNN

CNN induces a regularization effect through distinguish-ing between uppercase and lowercase letters(Zhang,Zhao, & LeCun, 2015).

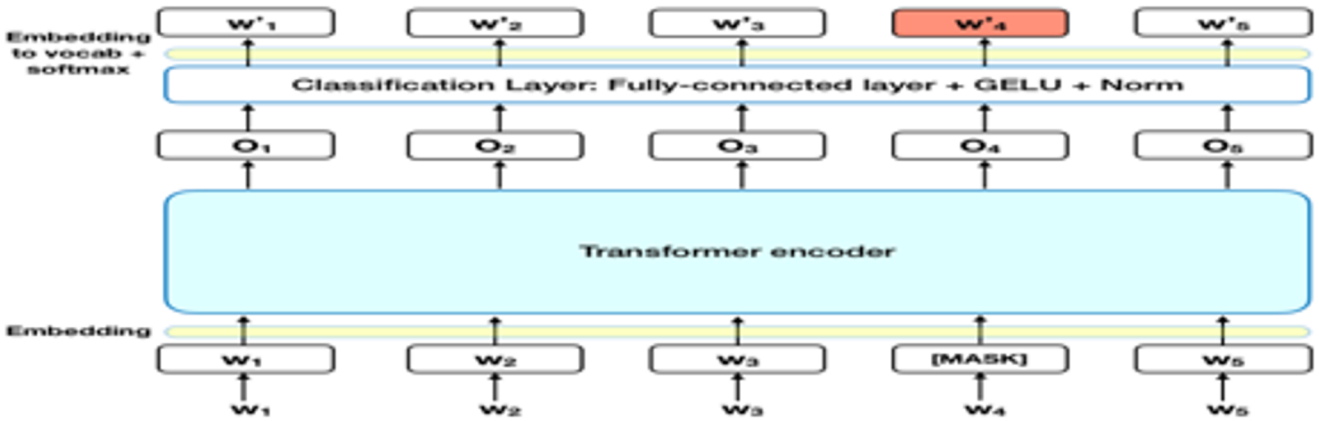
- The conventional method comes in handy in determining dichotomy in the semantic of tasks
- Dichotomy in task semantics does not play a major role in choosing better methods over CNN

- The approach adopts the simple use of a distributed word representation

Bidirectional Encoder Representations from Transformers (**BERT)**

BERT looks at text sequence from both left to right or combined left-to-right and right-to-left training

The approach learns contextual relations between words (or sub-words) in a text.



When It is BERT

- BERT facilitates learning the context of a word based on all of its surroundings

BERT uses the **Next Sentence Prediction (NSP)** and **Masked LM (MLM)** (Zhang, Zhao, & LeCun, 2015).

- **MLM** results to slower convergence and increased context awareness
- **NSP** uses the assumption that the random sentence will be disconnected from the first sentence.

Proposed Model Architecture

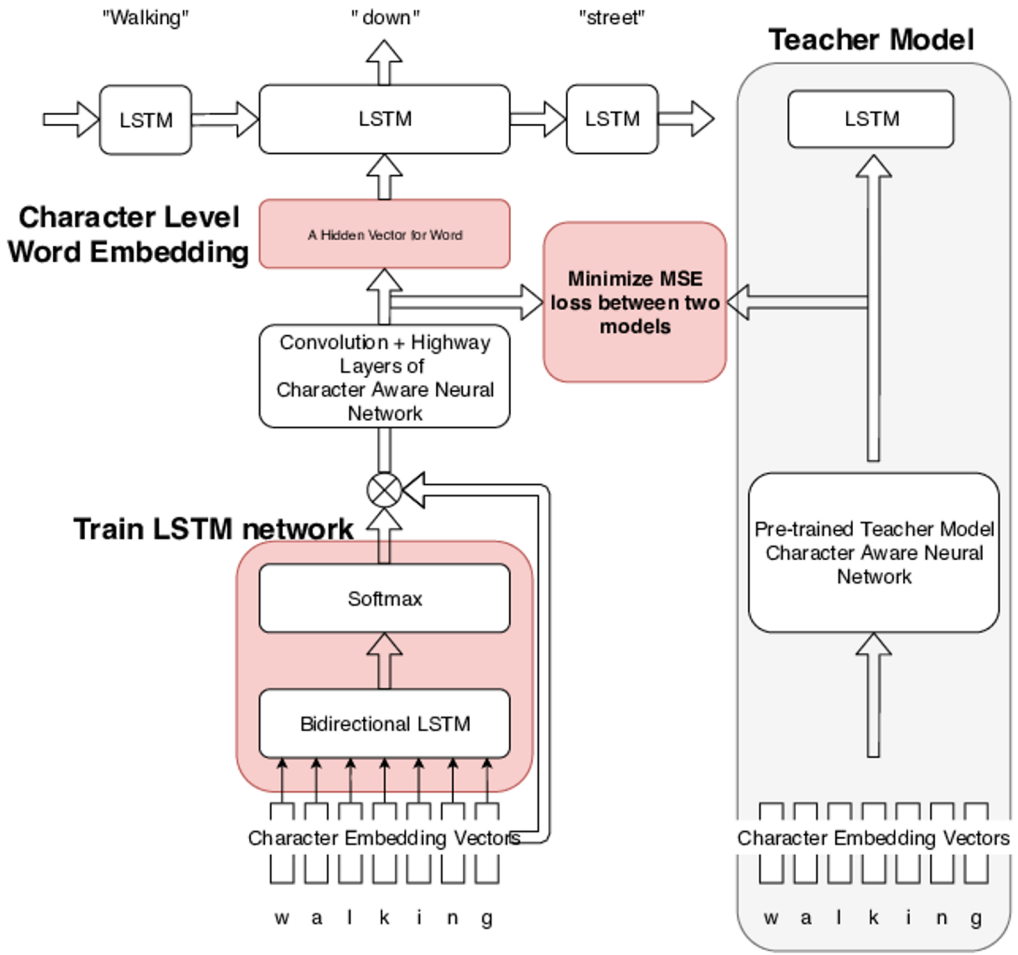
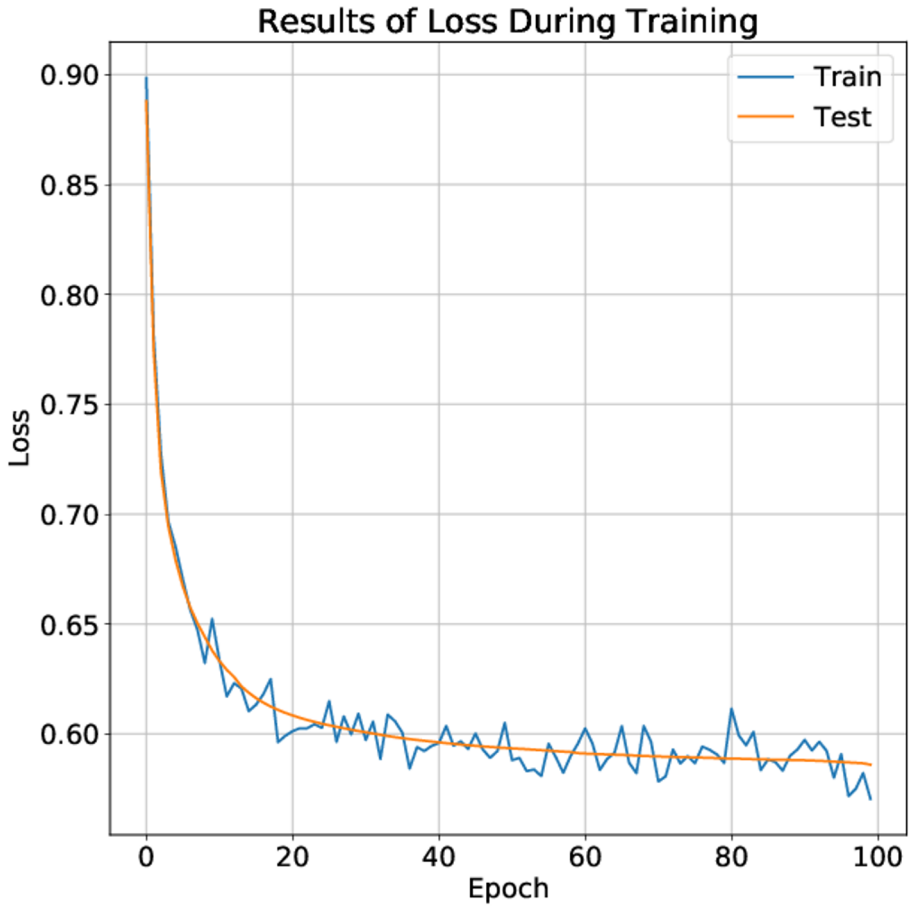
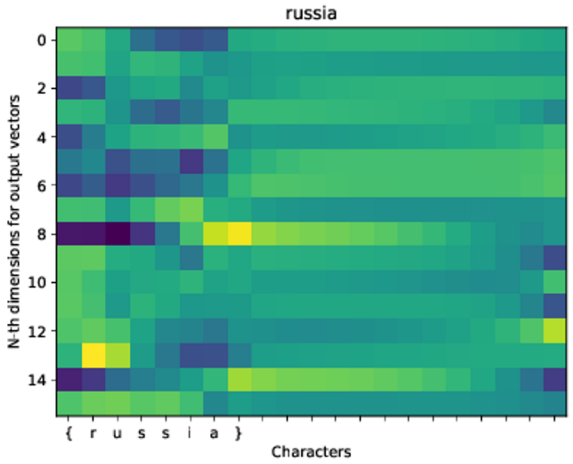


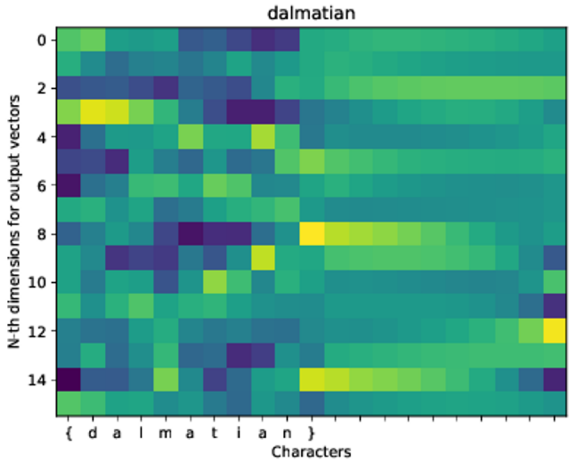
Fig. Model architecture

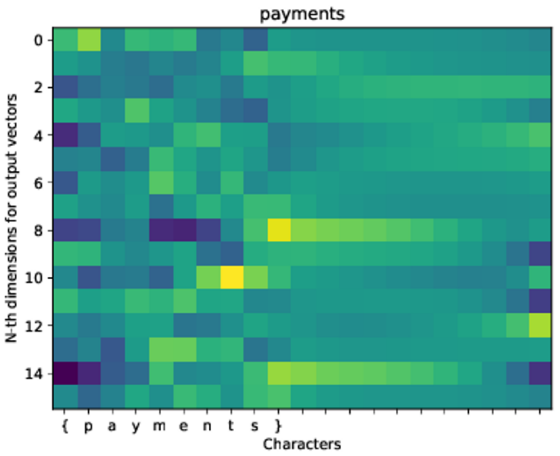
Results(training loss)

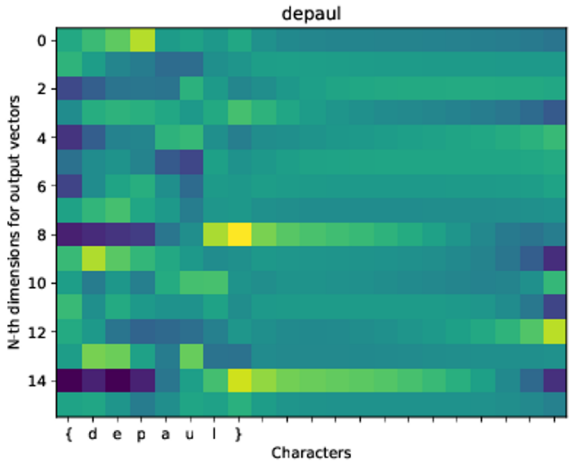


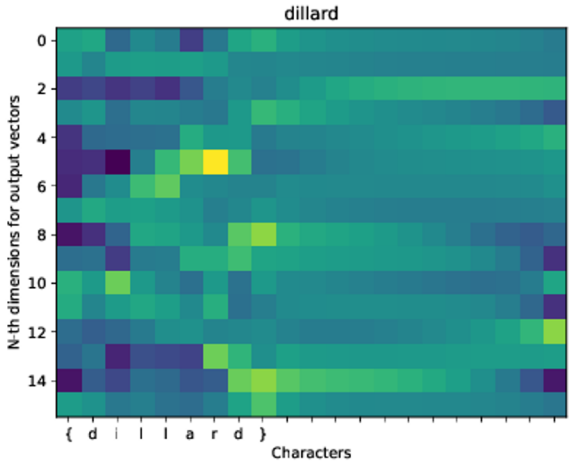
Results(visualization of gate map)

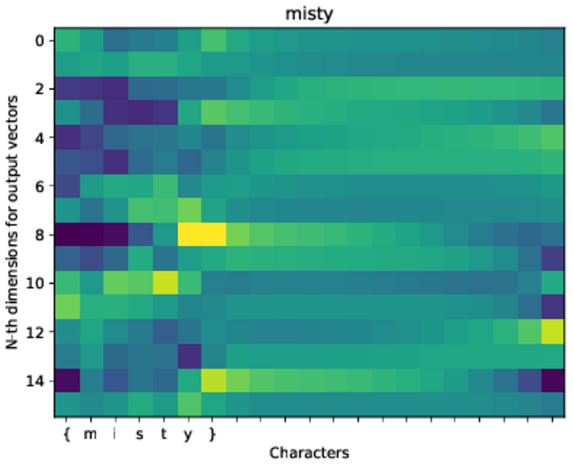












Results(closest words)

```
Top 7 words among our model: ['{degas}', '{vegas}', '{zetas}', '{deja}', '{dgbas}', '{pegasus}', '{aegis}']

Top 7 words between our model and Char CNN: ['{qf}', '{of}', '{oh}', '{q}', '{\\}', '{o}', '{488}']

Top 7 words among Char CNN: ['{degas}', '{vegas}', '{dgbas}', '{pegasus}', '{zetas}', '{banderas}', '{debakey}']

Top 7 words among word2vec(glove): ['{degas}', '{monet}', '{renoir}', '{manet}', '{cezanne}', '{matisse}', '{picasso}']




Top 7 words among our model: ['{contracted}', '{contractor}', '{contracts}', '{contract}', '{contractual}', '{contractors}', '{contradicted}']

Top 7 words between our model and Char CNN: ['{qf}', '{of}', '{oh}', '{q}', '{\\}', '{o}', '{488}']

Top 7 words among Char CNN: ['{contracted}', '{concentrated}', '{contradicted}', '{abstracted}', '{contrasted}', '{contacted}', '{contracts}']

Top 7 words among word2vec(glove): ['{contracted}', '{contracting}', '{accounted}', '{registered}', '{afflicted}', '{previously}', '{affected}']




Top 7 words among our model: ['{engine}', '{engineer}', '{engineering}', '{engineers}', '{engines}', '{engineered}', '{online}']

Top 7 words between our model and Char CNN: ['{of}', '{qf}', '{oh}', '{q}', '{\\}', '{o}', '{488}']

Top 7 words among Char CNN: ['{engine}', '{engineer}', '{engineering}', '{engineered}', '{engines}', '{online}', '{internecine}']

Top 7 words among word2vec(glove): ['{engine}', '{engines}', '{powered}', '{cylinder}', '{wheel}', '{turbine}', '{chassis}']




Top 7 words among our model: ['{formidable}', '{unforgivable}', '{avoidable}', '{sizable}', '{veritable}', '{recognisable}', '{readable}']

Top 7 words between our model and Char CNN: ['{of}', '{qf}', '{oh}', '{q}', '{\\}', '{o}', '{488}']

Top 7 words among Char CNN: ['{formidable}', '{unaffordable}', '{unsustainable}', '{unavoidable}', '{unforgivable}', '{sizable}', '{unpredictable}']

Top 7 words among word2vec(glove): ['{formidable}', '{fearsome}', '{daunting}', '{adversary}', '{tenacious}', '{dominating}', '{arguably}']
```

Conclusion

What we did

- We buit Bidirectional LSTM to implement a Gate for input character embeddings
- Gate is expected to reduce inputs

How were the results

- Loss is decreased
- Proposed network could find similar character word
- Proposed network could not find the morph gaps
- Proposed network could not learn the embedding of teaching model

How to improve

- We should add loss about second or third similar hidden layers
- We should implement mutual information into loss
- Character aware NN is not suited for learning because it is not good to find similar words

- word embedding distillation
  - regress Kim (2016)'s CharCNN without LSTM on BERT emeddings
    - subword segmented
    - static vs. contextual: Ethayarjah (2019)
- word embedding-based morphological segmentation
  - cosine similarity to detect morpheme boundaries (Üstün and Can, 2016): mt-caret/morpholine
  - deriving suffix/prefix morpheme translation rules from word embedding regularity (Soricut and Och, 2015)
- oov problem
  - solve via subword segmentation
    - BPE-based (Sennrich et al., 2016), subword regularization (Kudo, 2018), WordPiece (Google NMT) (Wu et al., 2016)
  - compose word embeddings from subword embeddings
    - fasttext (Bojanowski et al., 2017), Botha and Blunsom, 2014
      - fasttext-subword-interpreter model
  - subword attention
    - fasttext-subword-attention model

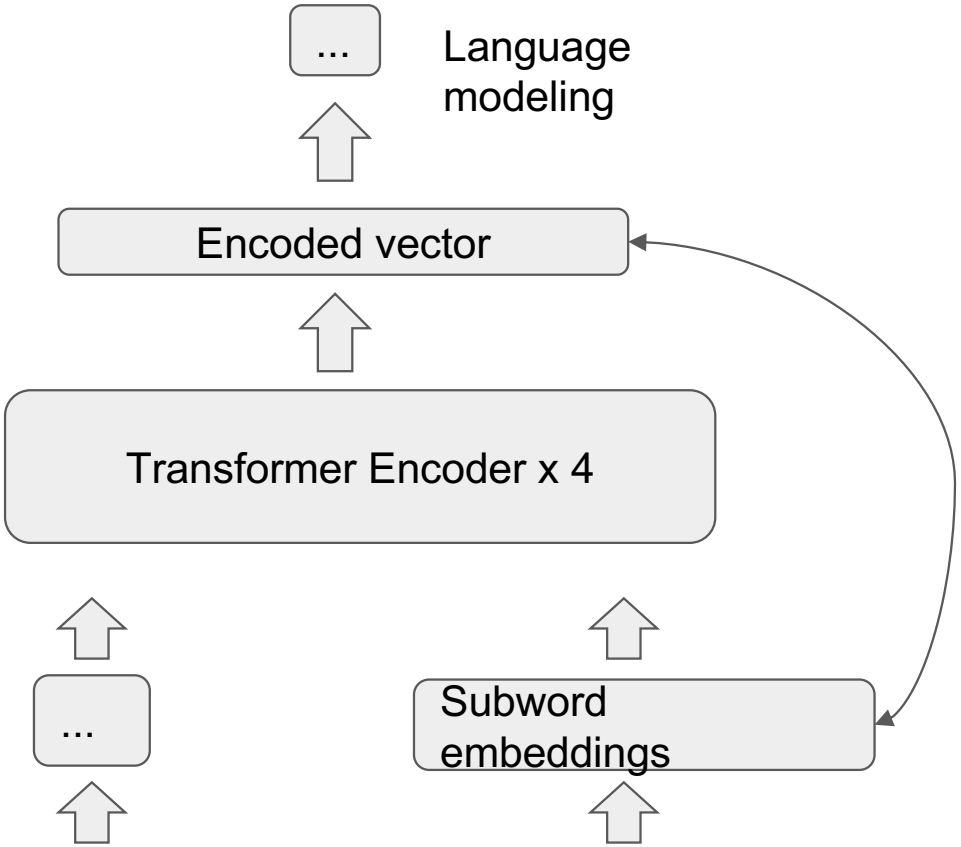
Morphological analysis using FastText

- Distillation from BERT easily converges to local minima
  - Learning word meaning from characters is difficult?
- Fasttext: Better trained model for morphological analysis?

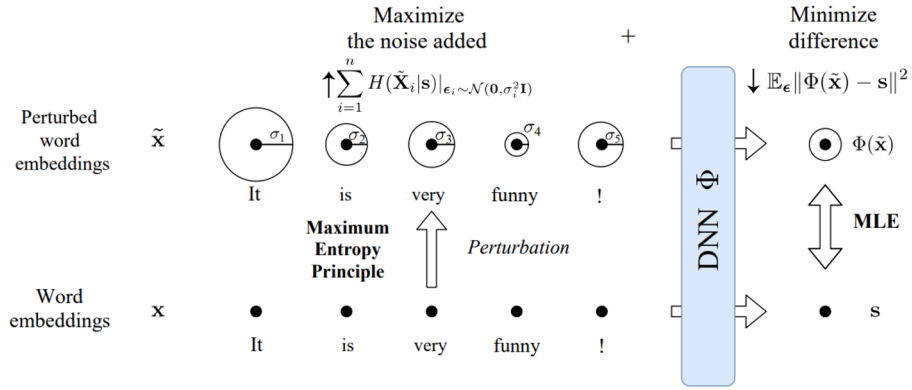
FastText

- Language modeling with subwords:
  - e.g. “where” → <wh, whe, her, ere, re>
  - Word embedding for each subwords
  - Train the language model with the subword embeddings

Morphological analysis using FastText

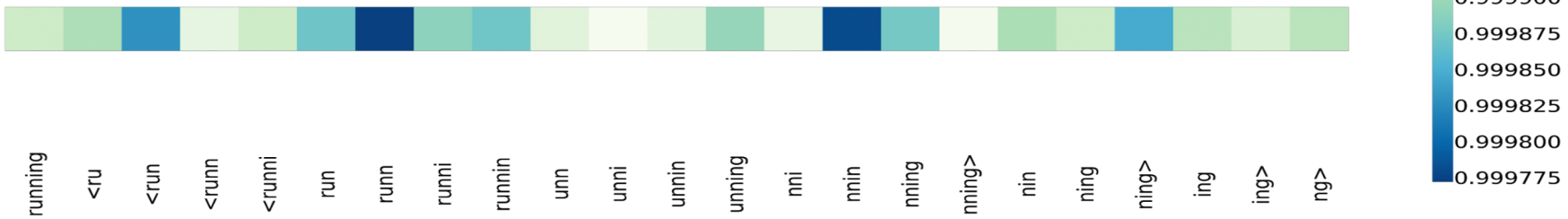


Mutual information analysis [Chaoyu et al. 2019]

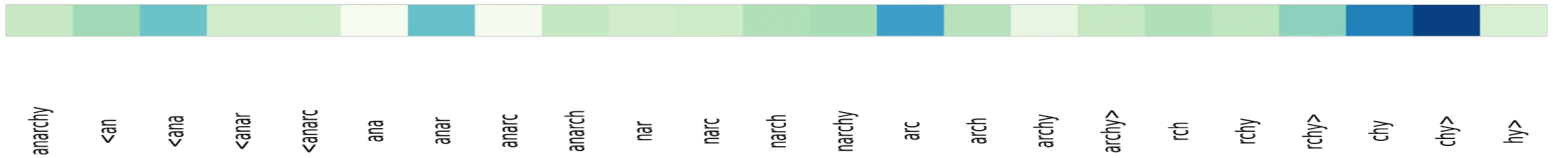


Results(FastText)

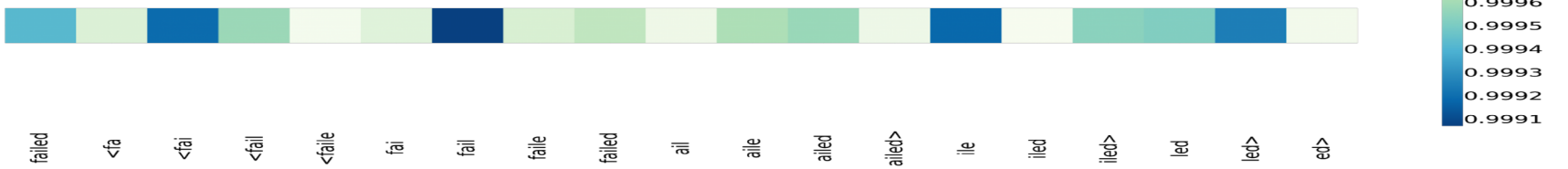
- “running”
  - Top 5 important words: ['runn', 'nnin', '<run', 'ning>', 'run']



- “anarchy”
  - Top 5 important words: ['chy>', 'chy', 'arc', 'anar', '<ana']



- “failed”
  - Top 5 important words: ['ed>', '<faile', 'iled']



Issues

- dataset size
- concerns with implementation
  - models don’t get that good at language modeling
  - interpreter implementation is iffy (sigma for words are very high, learning is unstable)