NLP Techniques that power the World of Words

Wednesday, June 12, 2019 10:46 AM

An unapologetic NLP fan boy view

This talk will be a semi-technical take on the evolution of techniques that has empowered NLP researchers on the analysis of languages. We will skim through the decades of progress but engorge heartily on the recent, exciting happenings in the world of words.

Agenda:

NLP Techniques that power the World of Words

0. Very Brief Intro to (WHY!!) NLP
1. Traditional NLP at a glance
2. Modern NLP
2A. Modern NLP :: Evolution of Language Models
2B. Modern NLP :: The 2 (quintessential) Word Embeddings
2C. Modern NLP :: DeepLearning Models for NLP
2D. Modern NLP :: Seq2Seq Models
2E. Modern NLP :: Pre-trained Language Models & Multi-task Learning
3. Open Problems/ Yet Unsatisfactory NLP Solution
References

0. Very Brief Intro to (WHY!!) NLP

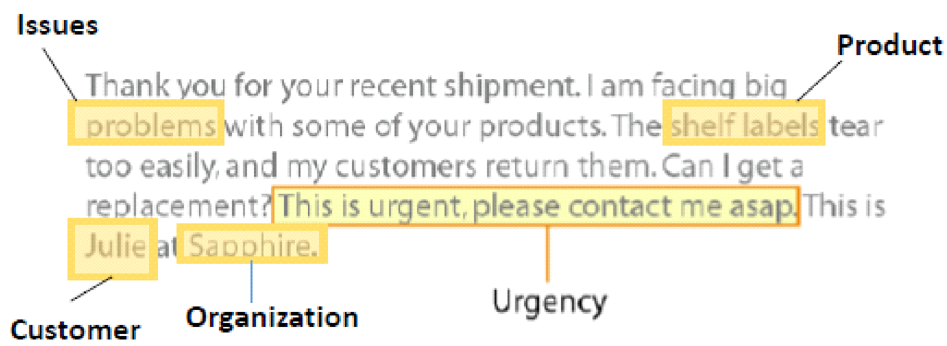
Saturday, June 15, 2019 3:07 PM

Not What is NLP, Why NLP?

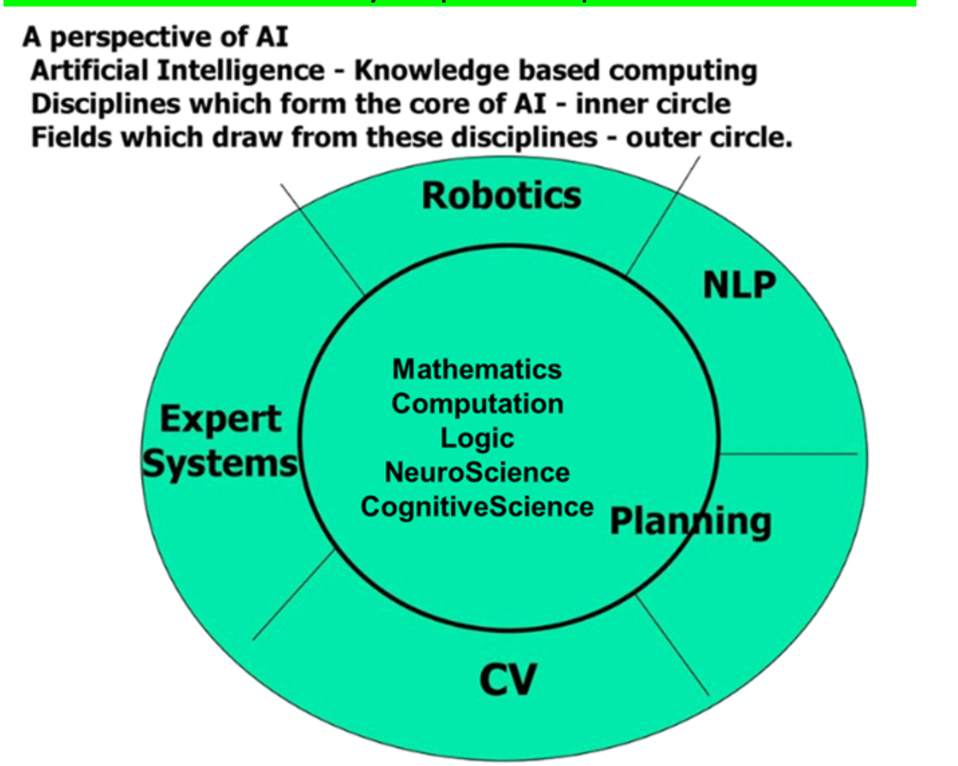
-- Lots of Text Data:

- In some cases, 80% of all enterprise data is unstructured text data
- Abundance of freely available text

-- Text packs a lot of information



-- NLP Growth - a very important part of AI evolution



*source: IIT Bombay
*Search - refers to Search Algorithms
*RSN-
*LRN - Local Response Normalization (a type of normalization technique similar to dropout or batch normalization)

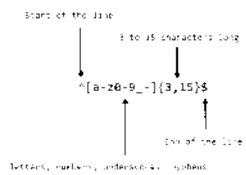
1. Traditional NLP at a glance

Saturday, June 15, 2019 3:45 PM

Computational Linguistic & Pattern Matching

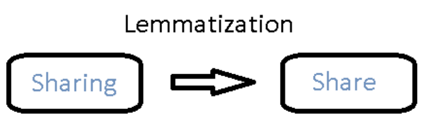
```
                 Sentence (S)
                      |
        +-------------+------------+
        |                          |
     Noun (N)              Verb Phrase (VP)
        |                          |
     John                 +-------+--------+
                          |                |
                       Verb (V)         Noun (N)
                          |                |
                       sees              Bill
```

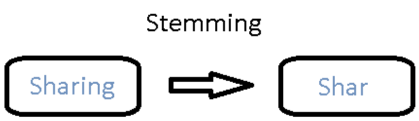

```
       root
        |
        | +-------dobj---------+
        | |                    |
 nsubj  | |   +------det-----+ | +-----nmod------+
 +--+   | |   |              | | |               |
 |  |   | |   |      +-nmod-+| | |      +-case-+ |
 +  |   + |   +      +      || + |      +      | |
 I  prefer  the  morning   flight  through  Denver
```



Traditional Statistical NLP Pre-processing

Stemming and Lemmatizing:





N-gram Analysis:

An Example of 3-*Gram*

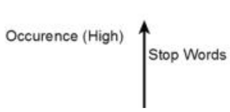
After sleeping for four hours, he decided to sleep for another four.

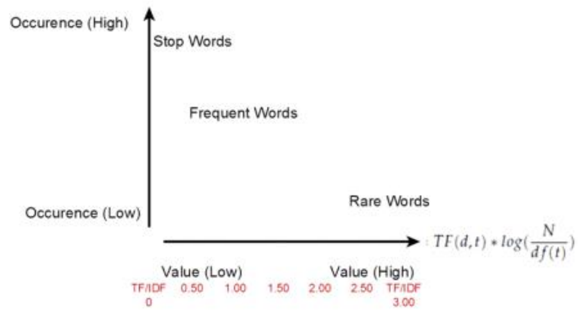
In this case, the tokens are as follows:

{ *"After sleeping for", "sleeping for four", "four hours he", " hours he decided", "he decided to", "to sleep for", "sleep for another", "for another four"* }.

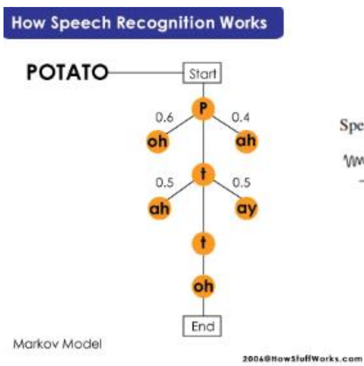
Bag-of-words representation concept:

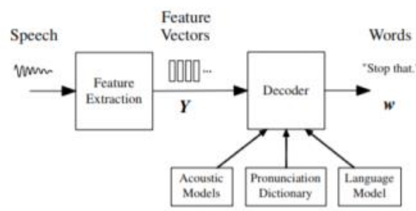
One e.g. of BOW: TF-IDF:





Traditional Statistical Probabilistic NLP Models





Given acoustic data $A = a_1, a_2, ..., a_k$

Find word sequence $W = w_1, w_2, ... w_n$

Such that $P(W \mid A)$ **is maximized**

Bayes Rule:

acoustic model (HMMs) language model

$$P(W \mid A) = \frac{P(A \mid W) \cdot P(W)}{P(A)}$$

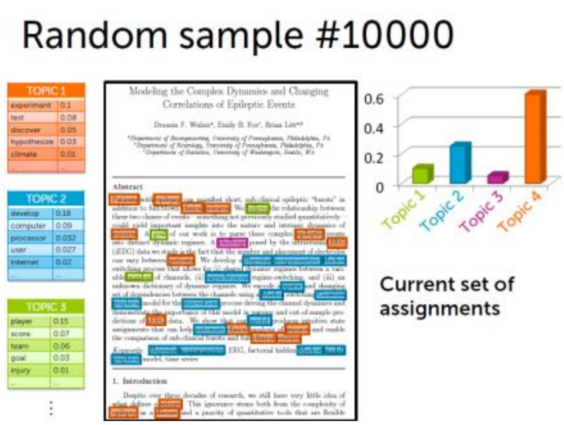
Condtional Random Fields

$$s(y, y', x) = \{\begin{matrix} 1 & iff(y = NOUN, y' = DET, x = "dog") \\ 0 & otherwise \end{matrix}$$

- Example: CRF POS tagging
  - Associates a tag (NOUN) with a word in the text ("dog") AND with a tag for the prior word (DET)
  - This function evaluates to 1 only when all three occur in combination
    - At training time, both tag and word are known
    - At evaluation time, we evaluate for all possible tag sequences and find the sequence with highest probability (Viterbi decoding)

26

Latent Dirichlet Allocation



2. Modern NLP

Wednesday, June 19, 2019 3:07 PM

Two important trends that power recent NLP growth

- Better **representation** of the text data (with no supervision)
    - By grasping the 'context' better
      (e.g. terms used language models, word embeddings, contextualized word embeddings)
    - By enabling 'transfer learning'
      (e.g.: term used pre-trained language models)

- **DeepLearning Models** that use the better representations to solve real-world problems like Text Classification, Semantic Role Labelling, Translation, Image Captioning, etc.,

2A. Modern NLP :: Evolution of Language Models

Wednesday, June 19, 2019 2:26 PM

Better Representation of Text

Topic_1: **Intro to Language Models**

- Task of **predicting the next word** given the previous words
- A language model can assign **probability to each possible next word**. And also, help in assigning a probability to an entire sentence.
- Arguably, the **simplest and most critical language processing task** with concrete practical applications
- Language modelling is a form of unsupervised, predictive learning --> no labeled text needed

Applications:

Predicting upcoming words or estimating probability of a phrase or sentence is useful in noisy, ambiguous text data sources

a. **Speech Recognition** - E.g.: P("recognize speech") >> P("wreck a nice beach")
b. **Spelling Correction** – E.g.: P("I have a gub") << P("I have a gun")
c. **Machine Translation** – E.g.: P("strong winds") > P("large winds")
d. **Optical Character Recognition/ Handwriting Recognition**
e. **Autoreply Suggestions**. E.g.: **Intelligent keyboards, auto email reply**
f. **Text Classification**

Topic_2: **Evolution of Language Models**

Sample Corpus:
This is **the house** that Jack built.
This is **the** malt
That lay in **the house** that Jack built.
This is the rat.
That ate **the** malt
That lay in **the house** that Jack built.
This is the cat.
That killed **the** rat.
That ate **the** malt
That lay in **the house** that Jack built.

P (house | the) = count (the house) / count (the)

Aside from the intuitive way we would calculate, this is an example of **Conditional Probability**!

P(B | A) = P(A , B) / P(A)

Technically, what we have computed above is **a Bigram Language Model**

$$Bigram\ model:\ p(w_t \mid w_{t-1})$$

Bi-gram LM
(generally, an N-gram LM)

Markov Bi-gram LM

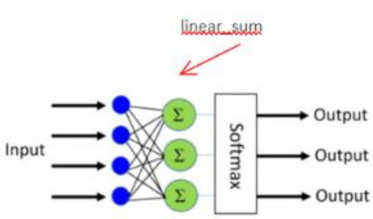
Hence the probability of occurrence of the 5-word sentence is:

$$p(A,B,C,D,E) = p(E \mid D)p(D \mid C)p(C \mid B)p(B \mid A)p(A)$$

****

If x is the current word vector and y is the next word vector,
Prob of y being the next word given the current word is x is given by

$$p(y \mid x) = softmax(W^T x)$$



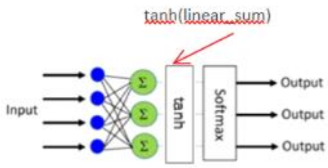
Equation of a softmax:

$$p(y = j \mid \mathbf{x}) = \frac{e^{(\mathbf{w}_j^T \mathbf{x} + b_j)}}{\sum_{k \in K} e^{(\mathbf{w}_k^T \mathbf{x} + b_k)}}$$

Where j is one of the K words

Logistic Bigram LM

****

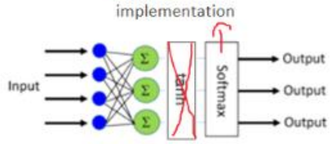


Feed Forward Neural Network LM

*proposed by Bengio et al in 2001

****

An approximate but efficient softmax implementation

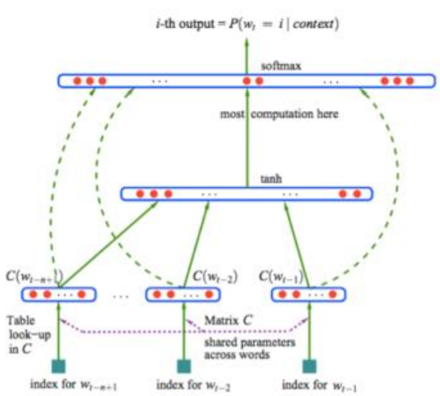


No activation

Word2Vec

*proposed by Mikilov et al in 2013

****

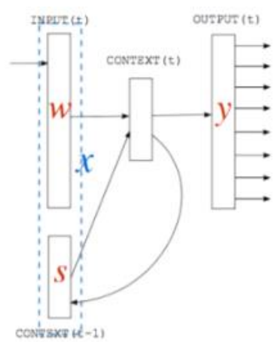


From Probabilistic Neural Language Models (Bengio - 2001)

-->

---->

****



$$x(t) = w(t) + s(t-1) \tag{1}$$

$$s_j(t) = f\left(\sum_i x_i(t)u_{ji}\right) \tag{2}$$

$$y_k(t) = g\left(\sum_j s_j(t)v_{kj}\right) \tag{3}$$

where $f(z)$ is sigmoid activation function:

$$f(z) = \frac{1}{1+e^{-z}} \tag{4}$$

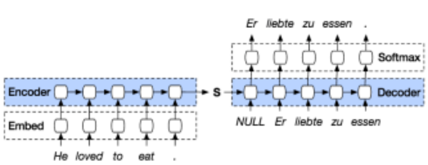
and $g(z)$ is softmax function:

$$g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}} \tag{5}$$

RNN-based Language Model (Mikilov - 2010)

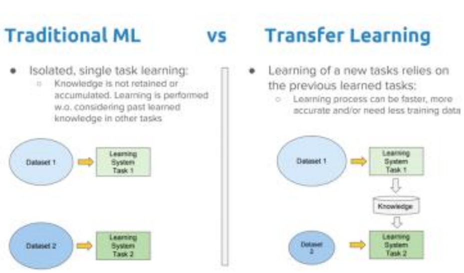
****

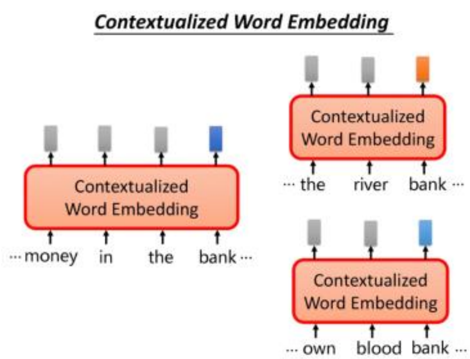
****



Seq2Seq Model

****





Contextualized Word Embedding provided by pre-trained Language Models

2B. Modern NLP :: The 2 (quintessential) Word Embeddings

Wednesday, June 19, 2019 3:01 PM

Better Representation of Text

Word Embeddings

W2v brought out relation between words such as gender, country-capital relations.

Matrix factorization techniques like LSA, SVD achieved the same result with proper(and extensive) tuning (and with good computational resources) -- thus came into being **glove**.

Let's dig a bit into the two typical Word Embeddings:

Word2Vec

Predict between every word and its context words!

Two algorithms

1. **Skip-grams (SG)**
   Predict context words given target (position independent)
2. Continuous Bag of Words (CBOW)
   Predict target word from bag-of-words context

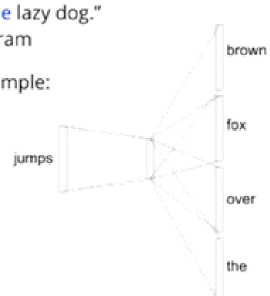
CBOW - continuous bag of words

"The quick brown fox jumps over the lazy dog."

Given a window size of 2 words around a focus word - jumps, **the CBOW model predicts the current word *'jumps',* given the neighboring words in the window**

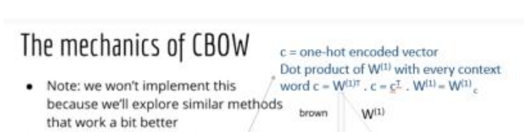
Skipgram

- "The quick brown fox jumps over the lazy dog."
- Helpful to think of it in terms of bigram
- Bigram model gives us 1 training sample:
  jumps → over
- Skipgram gives us 3 additional training samples:
  jumps → brown
  jumps → fox
  jumps → the



Given a window size of 2 words around a focus word, the **skip-gram model predicts the neighboring/context words given the current/focus word** (here – jumps).

1. How CBOW is different from a typical NN bigram model? - No activation function



Glove

The GloVe model learns word vectors by examining word co-occurrences within a text corpus.

- Before we train the actual model, we need to construct a co-occurrence matrix X, where a cell Xij is a "strength" which represents how often the word i appears in the context of the word j.

A simple window based co-occurrence matrix:

Example corpus:

- I like deep learning.
- I like NLP.
- I enjoy flying.

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

For example, for a window size of 3:

Context distance
"I love dogs and cats"
X(I, love) += 1
X(I, dogs) += ½
X(I, and) += ⅓

- We run through our corpus just once to build the matrix X, and from then on use this co-occurrence data in place of the actual corpus. We will construct our model based only on the values collected in X.

Long-tail distribution → non-zero values will be very large
So we will take the log
log X(i, j) will be the target
Add 1 before taking the log so we don't have NaNs

Range of X(i,j) will be from zero (no occurrence) to many occurrences.
Taking log X(i,j) will normalize this range.

We will produce embedding for each word wi and wj in the occurrence matrix such that it follows this equation

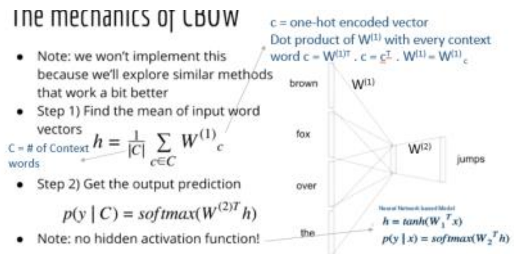
$$\vec{w}_i^T \vec{w}_j + b_i + b_j = \log X_{ij}$$

- Dot product of wi (1XK) and wj (1XK) = $w_i^T * w_j = w_i * w_i^T$ = (1X1) dimension
- bi and bj are scalar bias terms associated with words i and j, respectively

"GloVe is essentially a log-bilinear model with a weighted least-squares objective"

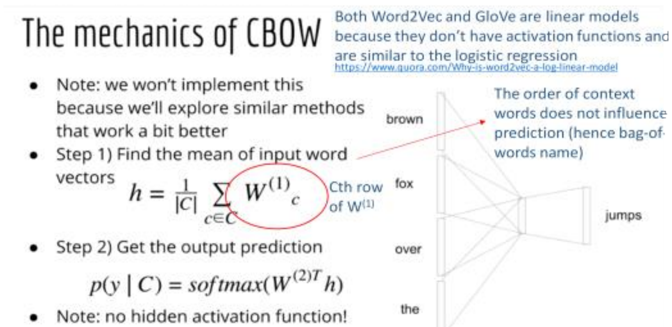
"global log-bilinear regression model which combines the benefits of both **global matrix factorization (Decompose large matrices into low-rank approximations)** and **local context window methods (Learn word representations using adjacent words.)**"

- A bilinear function (see https://en.wikipedia.org/wiki/Bilinear_map) is a function which



2. Compared to NN, Word2vec is a log-linear model

Neural Networks are non-linear because of activation function.



Why W2V log-linear?
"linear in log space"
To make training faster:

- Deep neural networks with **multiple stages of non-linear steps** are **slower to train** because during the backpropagation step we **need to propagate the gradient through each of the intermediate non-linear activation functions**.
- Word2vec being log-linear means we calculate the gradient at the output and then directly propagate this back into the embedding parameters (the main computational burden during training). This means faster trainer over bigger datasets yielding more accurate embedding vectors.

Conclusion on the basic theory:
Skip-gram: works well with small amount of the training data, represents well even rare words or phrases.
CBOW: several times faster to train than the skip-gram, slightly better accuracy for the frequent words.

approximations) and **local context window methods (Learn word representations using adjacent words.)**"

- A bilinear function (see https://en.wikipedia.org/wiki/Bilinear_map) is a function which is linear in each variable when all other variables are fixed. For instance, f(x,y) = x * y.
- **p(y | x) is log-linear if f(x,y) is linear in x and y. It is log-bilinear if f(x,y) is bilinear in x and y**.

Where it gets exciting? One particularly exciting direction is to project word embeddings of different languages into the same space to enable (zero-shot) cross-lingual transfer.

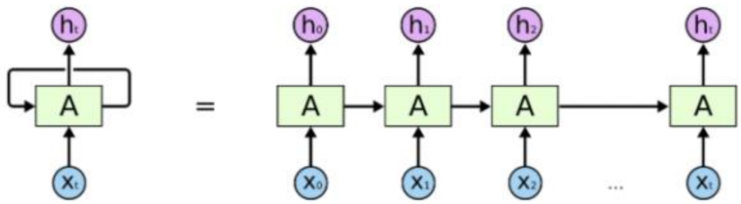
2C. Modern NLP :: **DeepLearning Models for NLP**

Wednesday, June 19, 2019 3:06 PM

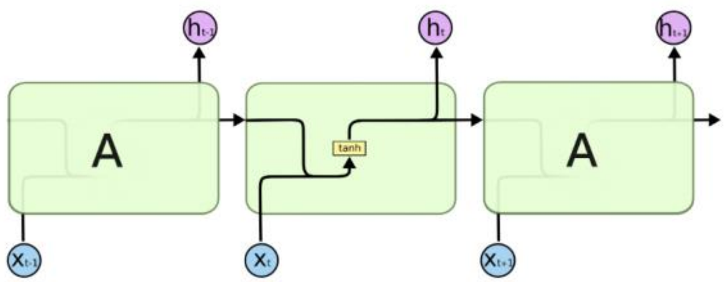
Deep Learning Models that serve better than ML models (supposedly in most situations)

1. RNN
2. CNN
3. Tree-based Recursive NN

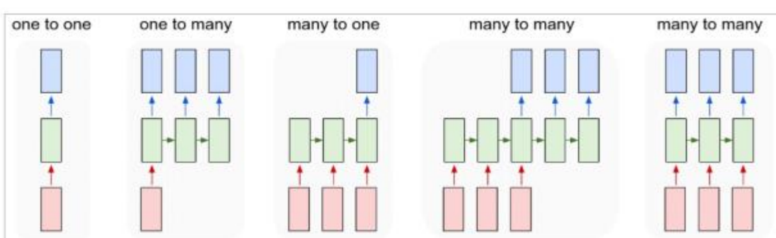
Recurrent neural networks (RNNs) are an obvious choice to deal with **the dynamic input sequences ubiquitous in NLP**.



An unrolled recurrent neural network.



Different types of RNN

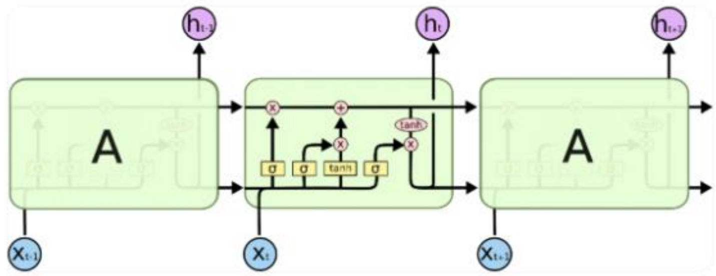


- Each rectangle is a vector
- Input (Red) | RNN's Unit (Green) | Output (Blue)
- The green recurrent units can be 'applied' as many times as we like

(1) - Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification)
(2) - Sequence output (e.g. image captioning takes an image and outputs a sentence of words).
(3) - Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment).
(4) - Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French).
(5) - Synced sequence input and output (e.g. video classification where we wish to label each frame of the video).

One important limitation of RNN - Vanishing Gradient.

Solution - (long-short term memory networks) LSTMs

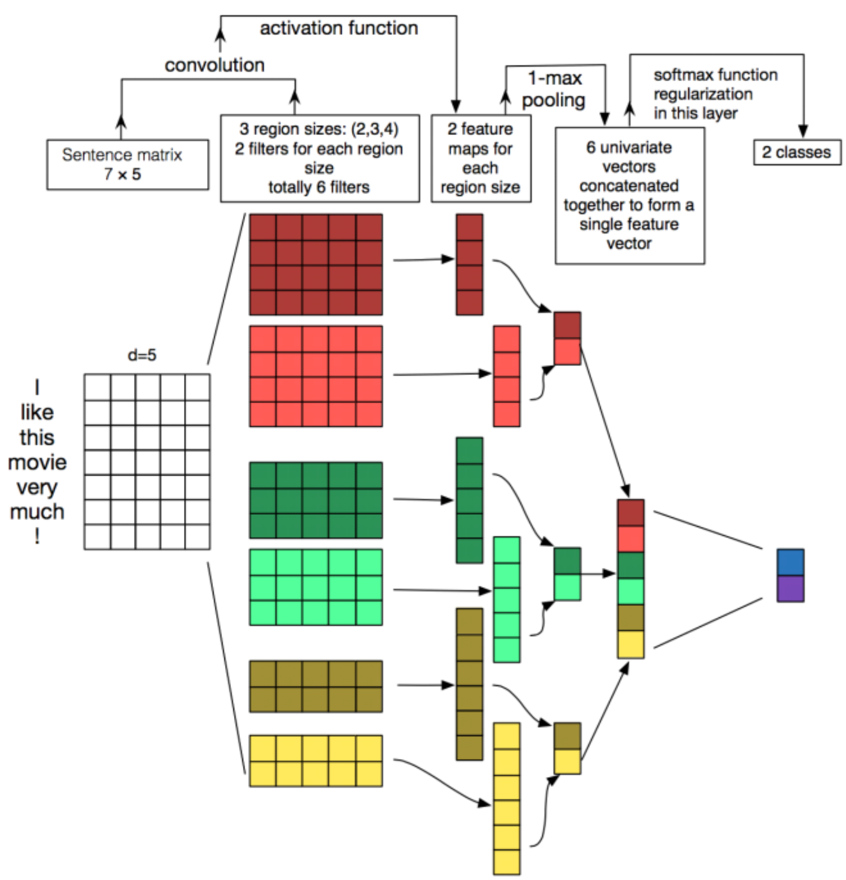


proved more resilient to the Vanishing and Exploding Gradient problems in RNN.
-- Hence wherever vanilla RNN was used, got replaced with LSTM

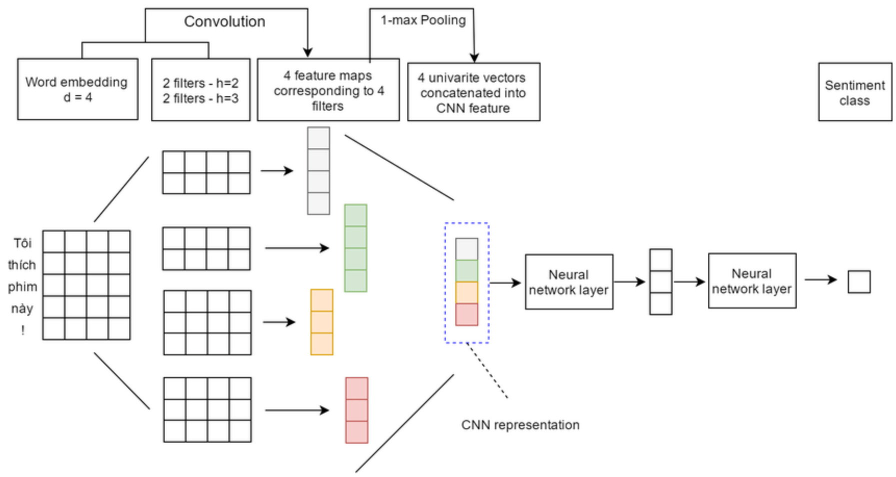
<u>CNNs for Text</u>

CNN for text

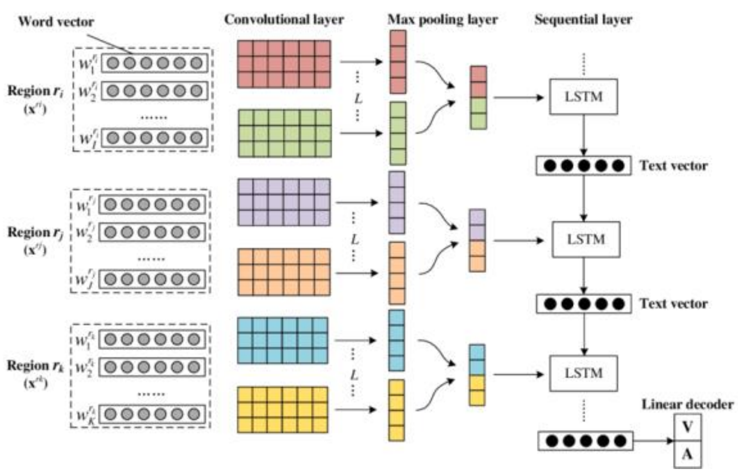
- The Sentence Matrix consists of features (list of words or even characters representing the vocabulary) in rows
- The columns of the Sentence Matrix could be
    - One-hot representation of vectors
    - Dimensions of Word2Vec or Glove vector
    - The filter size will always be <user_given_filter_size_hyperparameter> x <dimension_of_the_feature>



CNN + 2 layers of NN (any)



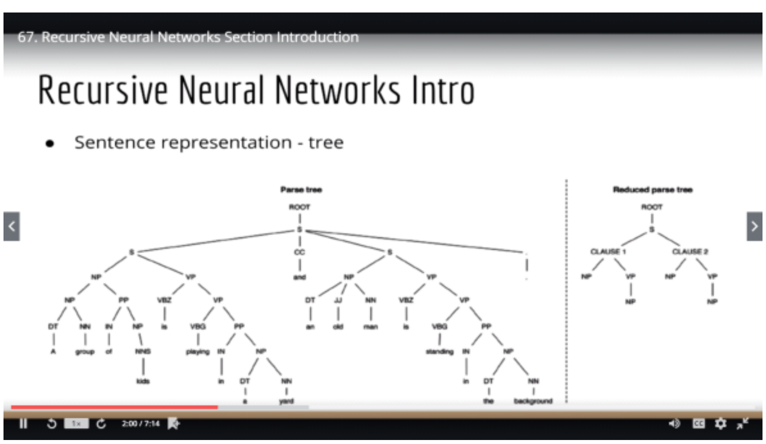
CNN + Sequential Layer



<u>Recursive NN</u>:

- Long term dependencies are tougher to capture for RNNs (even for LSTMs and the ones more advanced like Attention, Transformer, etc.,)
- We humans do not retain in memory a long sentence from left to right as in a sequence. Instead we compartmentalize and save

Idea of recursive neural networks - treat natural language as a hierarchical tree and not as a sequence.

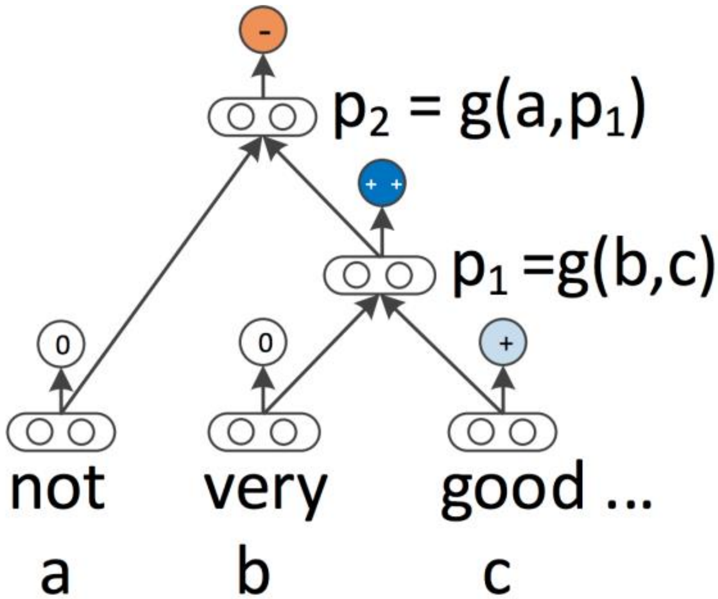


Recursive neural networks build the representation of a sequence from the bottom up in contrast to RNNs who process the sentence left-to-right or right-to-left. At every node of the tree, a new representation is computed by composing the representations of the child nodes

Cross-combination uses of these Tree NN:

- LSTM units can be used for Recursive NN
- **Word embeddings** can be learned based not only on local but on grammatical context (Levy & Goldberg, 2014)
- **language models can generate words based on a syntactic stack** (Dyer et al., 2016)
- **graph-convolutional neural networks** can operate

Below image from **a recursive neural network** (Socher et al., 2013)



2014)

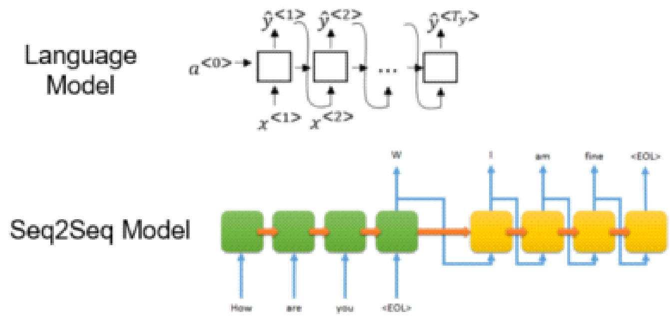
- **language models can generate words based on a syntactic stack** (Dyer et al., 2016)
- **graph-convolutional neural networks** can operate over a tree (Bastings et al., 2017)

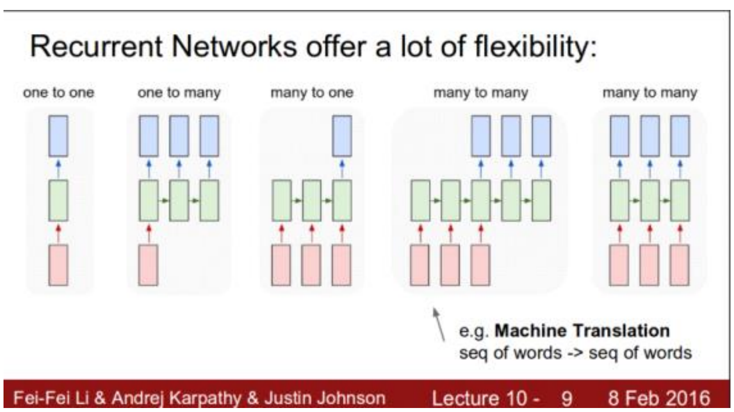
2D. Modern NLP :: Seq2Seq Models

Wednesday, June 19, 2019 3:20 PM

Seq2Seq Models -- A logical extension of Language Models marrying better NN units

Evolution of Seq2Seq Models from Language Models

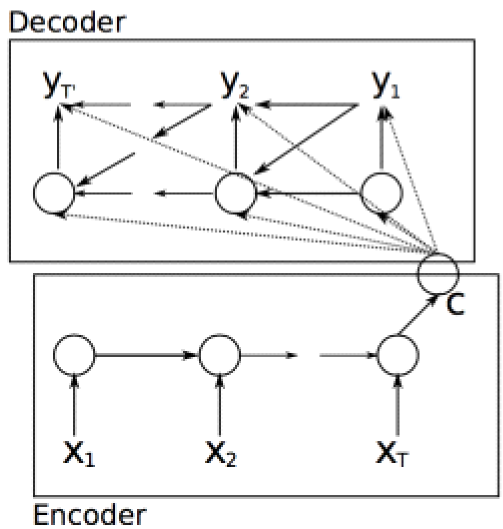




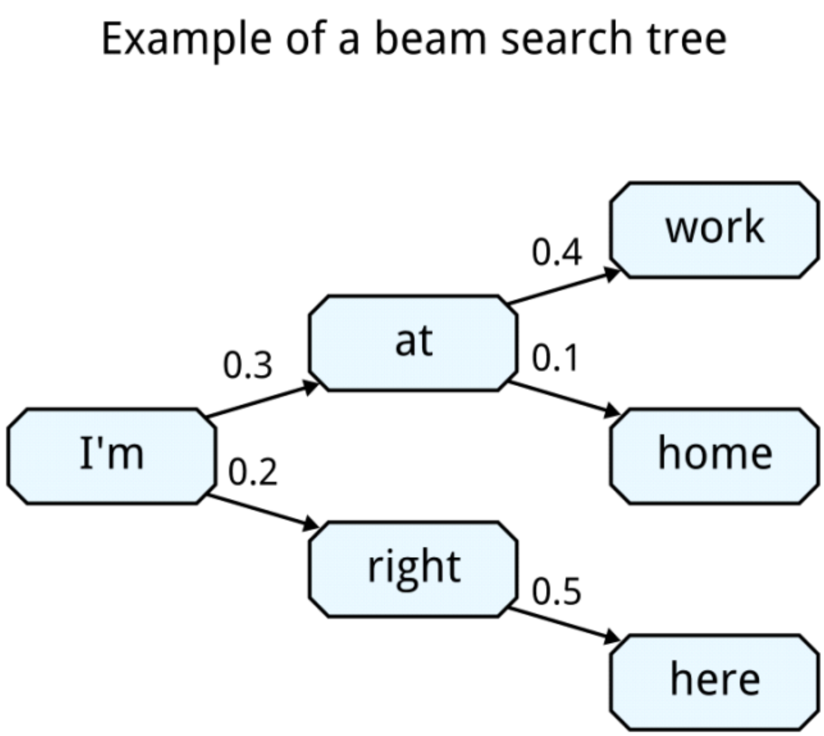
Normal Seq2Seq Models

Encoders and Decoders are 2 different neural networks joined together

- the task of an encoder network is to understand the input sequence, and create a smaller dimensional representation of it - called 'Context' or 'Thought' vector



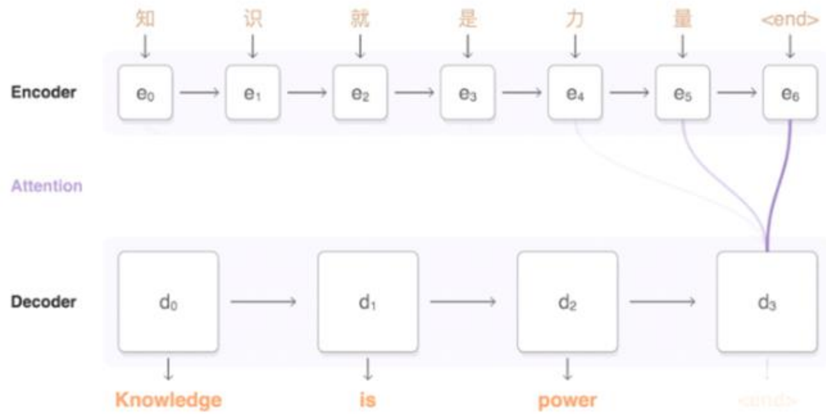
Use of **Beam Search Tree** in the decoder side to get the best output sequence



One main limitation of Seq2Seq Models:

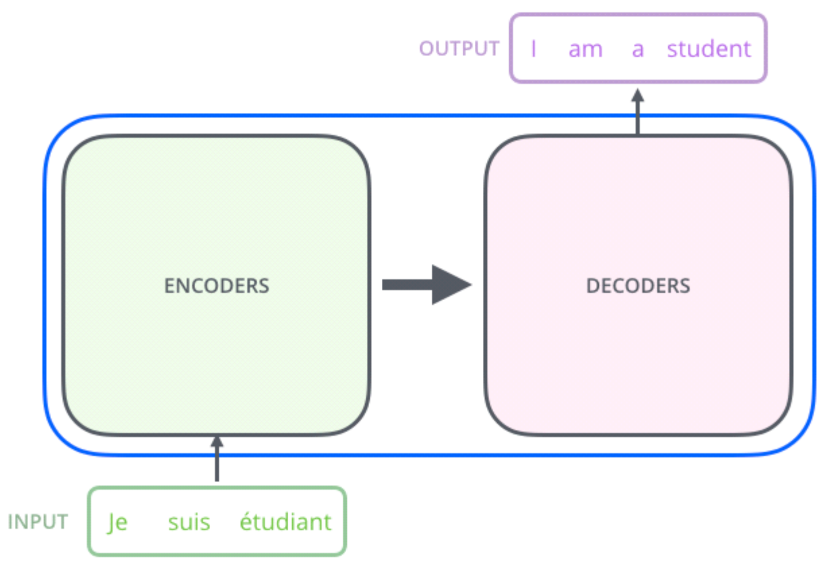
- sequence-to-sequence learning is that it requires to compress the entire content of the source sequence into a fixed-size vector

Attention solves this limitation by allowing the decoder to look back at the source sequence hidden states, which are then provided as a weighted average as additional input to the decoder

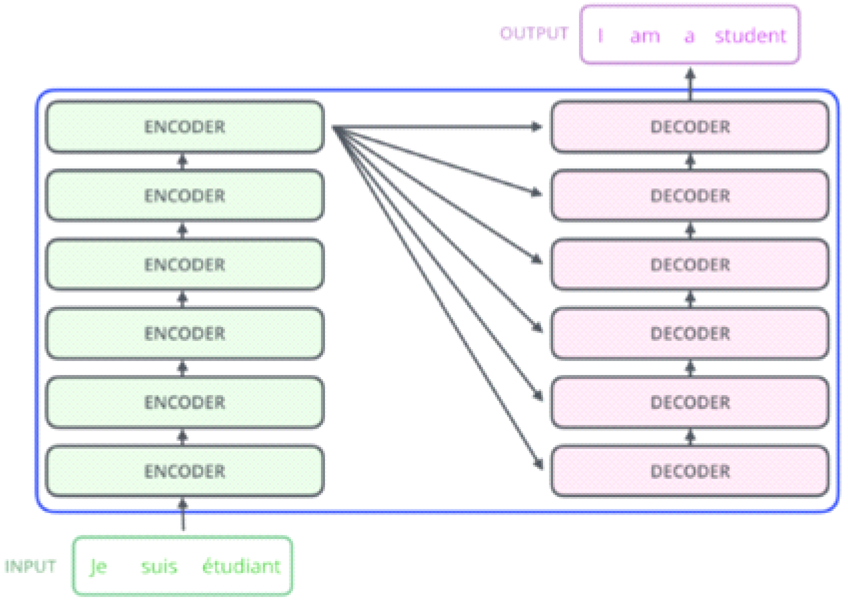


- **Self-attention**:
- self-attention can be used to look at the surrounding words in a sentence or document to obtain more contextually sensitive word representations

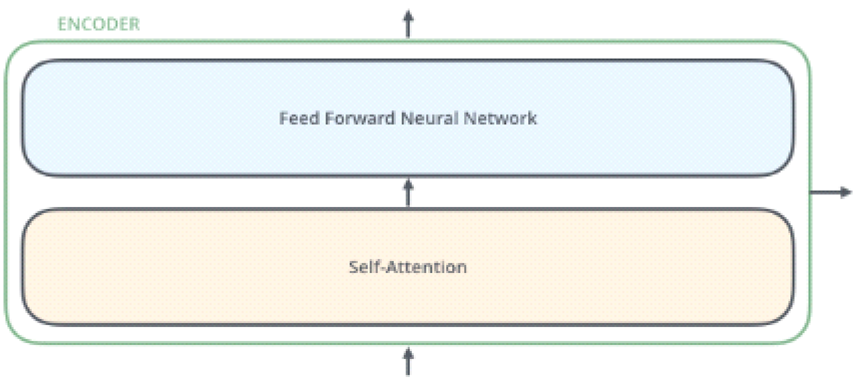
Multiple layers of Self-Attention forms the core of **Transformer** architecture:



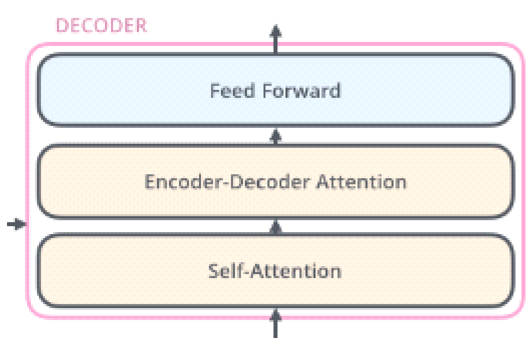
Transformer:



Encoder:



Decoder:

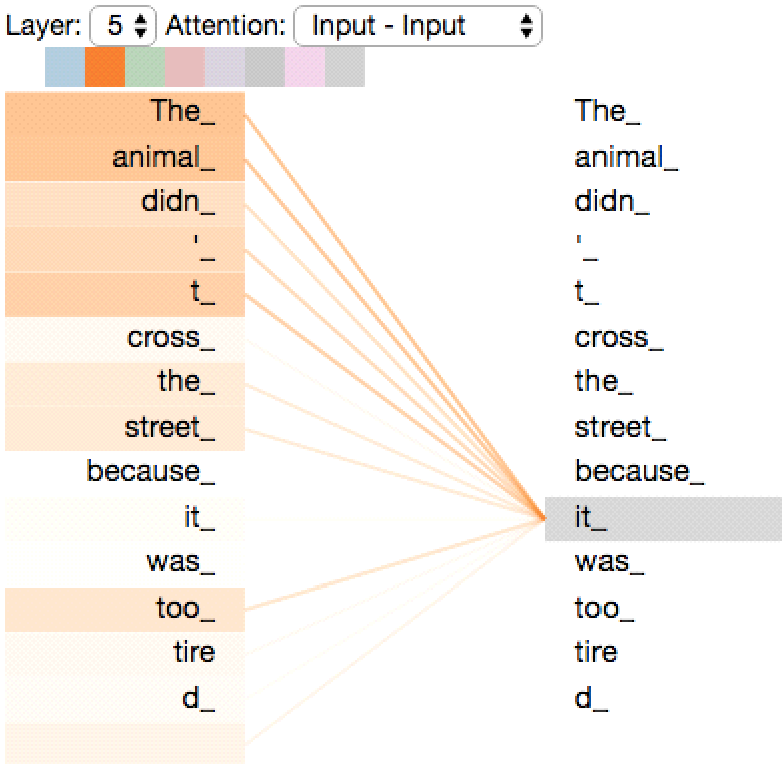


What is self-attention?

The sentence we want to translate?

"`The animal didn't cross the street because it was too tired`"

Self-attention is the method the Transformer uses to bake the "understanding" of other relevant words (e.g.: 'The', 'animal') into the one (e.g.: 'it') we're currently processing



==> Attention == Fuzzy Memory ?!
==> Doesn't work that well on longer sequences!

Attention can be seen as a form of fuzzy memory where the memory consists of the past hidden states of the model, with the model choosing what to retrieve from memory.

Memory networks have more explicit memory.

Memory-based models are typically applied to tasks, where retaining information over longer time spans should be useful such as language modelling and reading comprehension.

2E. Modern NLP :: Pre-trained Language Models & Multi-task Learning

Wednesday, June 19, 2019 3:22 PM

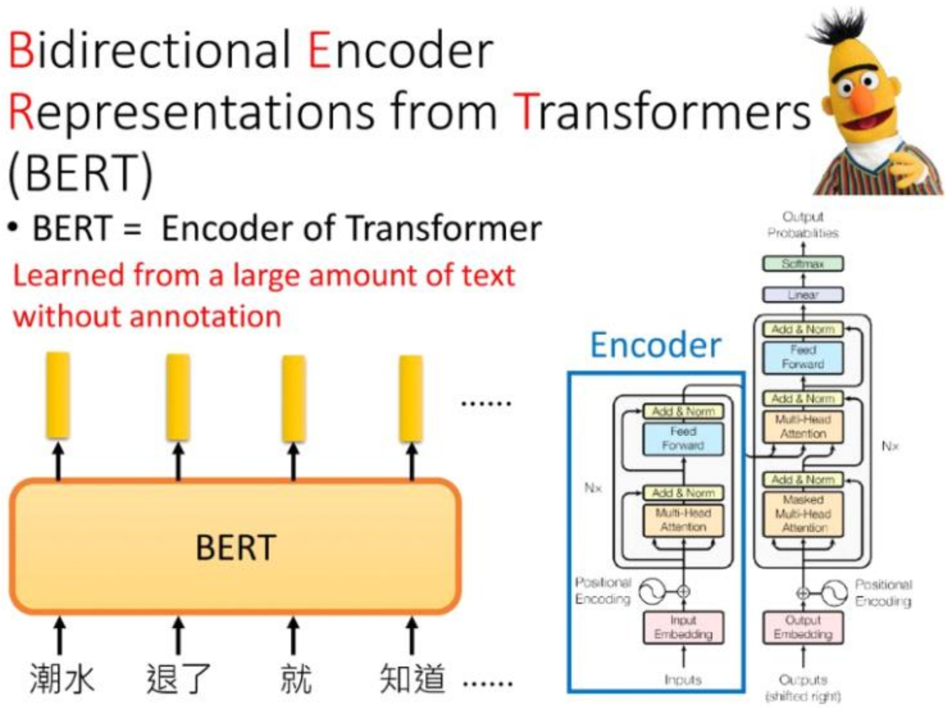
Better Representation of Text

Multi-task Learning

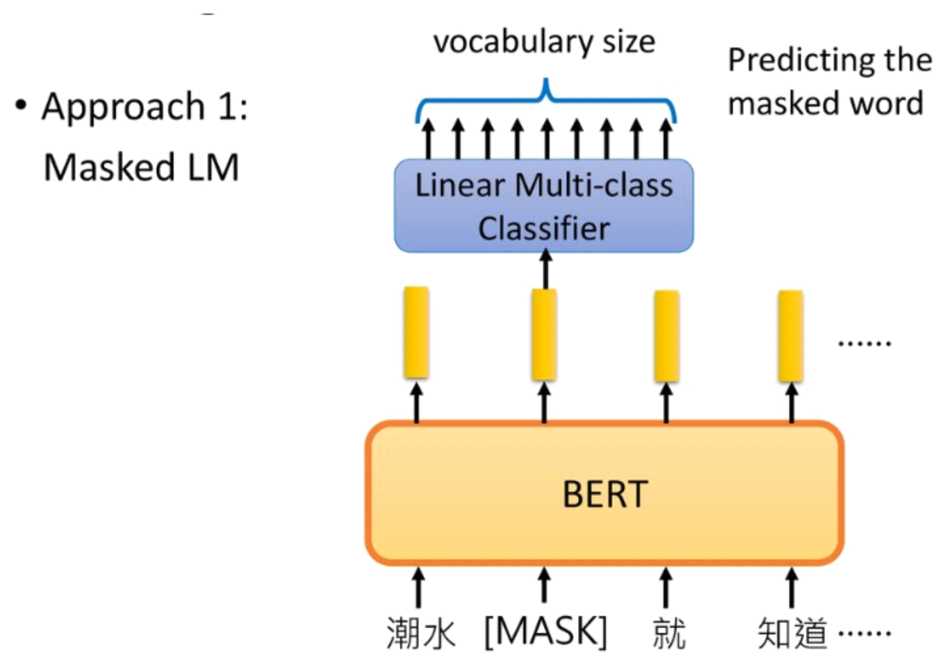
Motivation -
In the movie The Karate Kid (1984), sensei Mr Miyagi teaches the karate kid seemingly unrelated tasks such as sanding the floor and waxing a car. In hindsight, these, however, turn out to equip him with invaluable skills that are relevant for learning karate.

Multi task model share parameters

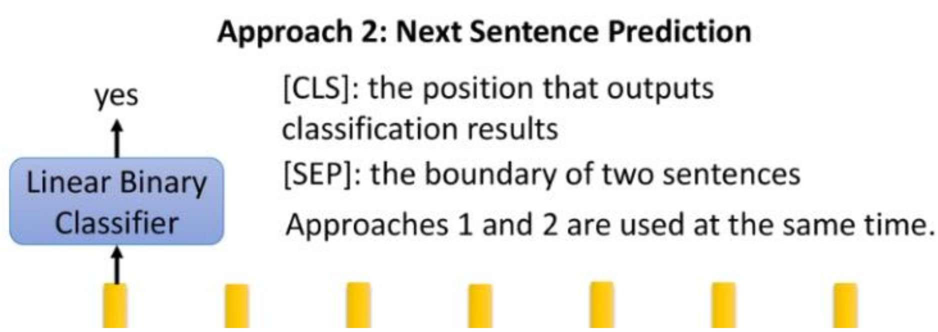
E.g. of
multi task learning + Transformer :
Bert - a pretrained Language modelling representation that is originally trained to do two tasks

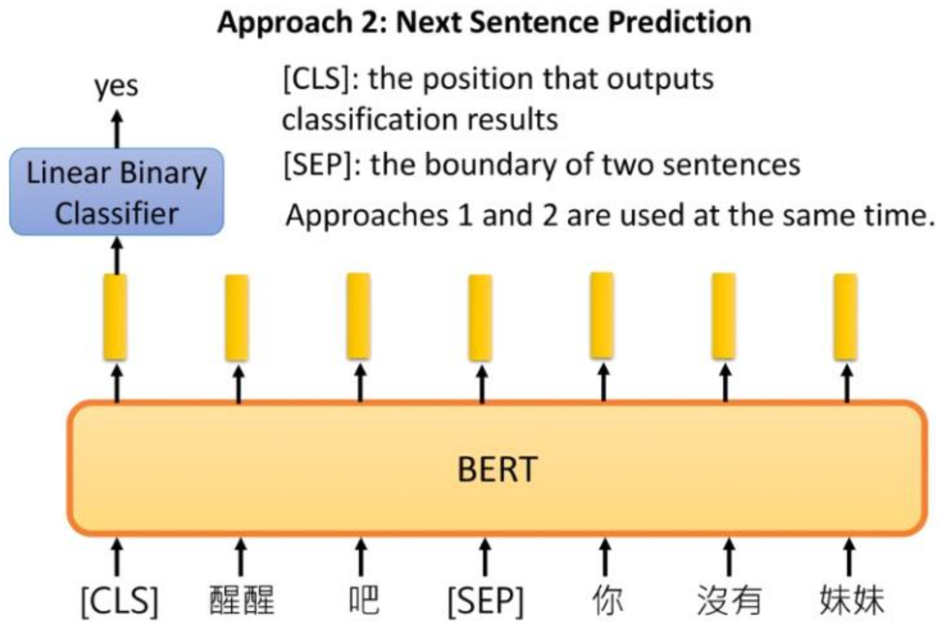


1. LM (specifically called masked LM)



2. Next Sentence Prediction





Simply put,

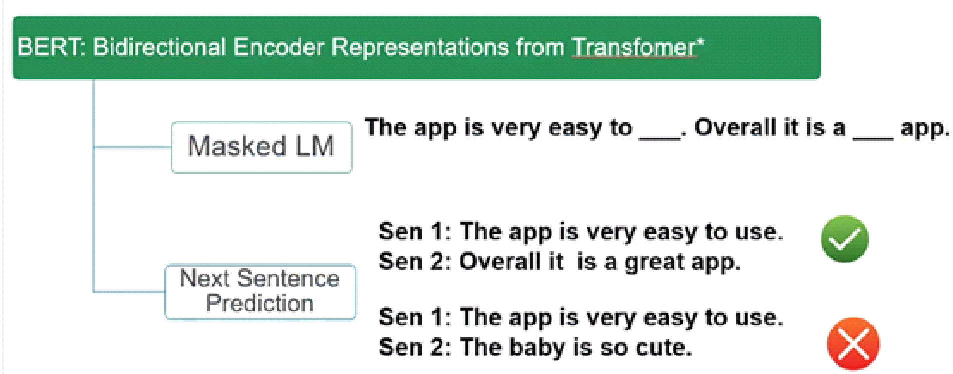


*image courtesy: Xiaopei & Josh's upcoming presentation on Fine-grained sentiment

3. Open Problems/ Yet Unsatisfactory NLP Solution

Wednesday, June 12, 2019 11:54 AM

Human-quality fake text is already here !!!! OpenAI GPT2 already bettering the SOTA BERT!

SYSTEM PROMPT (HUMAN-WRITTEN)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES)

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them – they were so close they could touch their horns.

While examining these bizarre creatures the scientists discovered that the creatures also spoke some fairly regular English. Pérez stated, "We can see, for example, that they have a common 'language,' something like a dialect or dialectic."

But still there are 'OPEN' problems:

Problem 1: Natural language understanding

I think the biggest open problems are all related to natural language understanding... **we should develop systems that read and understand text the way a person does**,

To achieve NLU, is it important to build models that process language "the way a person does"?

How do you think we would go about doing this?

Do we need inductive biases or can we expect models to learn everything from enough data?

Problem 2:

NLP for low-resource scenarios

> Dealing with low-data settings (low-resource languages, dialects (including social media text "dialects"), domains, etc.). This is not a completely "open" problem in that there are already a lot of promising ideas out there; **but we still don't have a universal solution to this universal problem**.
> – Karen Livescu

Generalisation beyond the training data – relevant everywhere!

Domain-transfer, transfer learning, multi-task learning

Learning from small amounts of data

Semi-supervised, weakly-supervised, "Wiki-ly" supervised, distantly-supervised, lightly-supervised, minimally-supervised

Unsupervised learning

Problem3:

Reasoning about large or multiple documents

> Representing large contexts efficiently. **Our current models are mostly based on recurrent neural networks, which cannot represent longer contexts well.** [...] The stream of work on graph-inspired RNNs is potentially promising, though has only seen modest improvements and has not been widely adopted due to them being much less straight-forward to train than a vanilla RNN.
> – Isabelle Augenstein

Problem4:

Datasets, problems, and evaluation

> Perhaps the biggest problem is to **properly define the problems themselves.** And by properly defining a problem, I mean building **datasets and evaluation** procedures that are appropriate to measure our progress towards concrete goals. Things would be easier if we could reduce everything to Kaggle style competitions!
> – Mikel Artetxe

References

Wednesday, June 19, 2019 3:43 PM

Useful Material References:
Main Source for idea:

- http://blog.aylien.com/a-review-of-the-recent-history-of-natural-language-processing/
- Any definition related to NLP Terms:
  https://github.com/sebastianruder/NLP-progress/tree/master/english

Pic References:

0.

- Data Deluge pic -- Economist Magazine
- Text packs a lot of information pic - Slide by Karthikeyan Sanakarn, Dir @ LatentView
  https://github.com/skkeyan-mlai/NLP/blob/master/Natural%20Language%20Processing%20-%20Motivation%20%26%20Mechanics%20-%20By%20Karthikeyan%20Sankaran.pdf
- NLP in AI:
  https://slideplayer.com/slide/13603947/
  https://www.cs.bham.ac.uk/~jxb/IAI/w2.pdf

1.

- Constituency Parsing:
  http://nlpprogress.com/english/constituency_parsing.html
- Dependency Parsing :
  https://nlpforhackers.io/complete-guide-to-spacy/
  http://nlpprogress.com/english/dependency_parsing.html
- HMM for Speech Recognition:
  https://electronics.howstuffworks.com/gadgets/high-tech-gadgets/speech-recognition2.htm
  Pic of the HMM-based Speech Recognizer:
  https://mi.eng.cam.ac.uk/~mjfg/mjfg_NOW.pdf
  MM vs HMM:
  http://www.cs.cmu.edu/~roni/10601-slides/hmm-for-asr-whw.pdf

- CRF for POS Tagging (Ohio State University):
  https://slideplayer.com/slide/4136134/
- Intro to CRF:
  https://www.research.ed.ac.uk/portal/files/10482724/crftut_fnt.pdf
- Difference between HMM and CRF:
  https://stats.stackexchange.com/questions/58221/intuitive-difference-between-hidden-markov-models-and-conditional-random-fields

- Latent Dirichlet Allocation snapshot from Coursera-Washington

2.

- Bigram LM, Logistic LM, NN LM - Snapshots from Udemy course on NLP by Lazy Programmer
- Probabilistic NN LM:
  http://ruder.io/word-embeddings-1/index.html#classicneurallanguagemodel
- RNN LM:
  https://www.cs.cmu.edu/~hiroakih/pdf/RNNLM_hiroakih.pdf
- Contextualized Word Embedding:
  https://slideplayer.com/slide/17025344/
- Word2Vec and Glove - Snapshots from Udemy course on NLP by Lazy Programmer

- RNN, unrolled RNN, different types of RNN
  https://colah.github.io/posts/2015-08-Understanding-LSTMs/
  http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- CNN for text
  https://medium.com/jatana/report-on-text-classification-using-cnn-rnn-han-f0e887214d5f
  http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/
- Recursive Neural Network - Snapshots from Udemy course on NLP by Lazy Programmer
- **A recursive neural network** (Socher et al., 2013)
  http://blog.aylien.com/a-review-of-the-recent-history-of-natural-language-processing/
- Attention Illustration, Self-attention explanatory pics
  http://jalammar.github.io/illustrated-transformer/
- BERT pics
  https://slideplayer.com/slide/17025344/

3.
- GPT2:
  https://blog.floydhub.com/gpt2/
- Open NLP Problems:
  https://www.slideshare.net/SebastianRuder/frontiers-of-natural-language-processing (slide 45)