Sequencing Legal DNA

NLP for Law and Political Economy

10. Text Generators

Outline

Language Modeling

- “Language Modeling” is a somewhat confusing label for the task of teaching an algorithm to predict/generate language.

Language Modeling

- "Language Modeling" is a somewhat confusing label for the task of teaching an algorithm to predict/generate language.
- The standard approach uses the Markov assumption: future words are independent of the past given the present and some finite number of previous rounds.
  - A kth order markov-assumption assumes that the next word in a sequence depends only on the last k words:

$$\Pr(w_{i+1} | w_{1:i}) \approx \Pr(w_{i+1} | w_{i-k:i})$$

Language Modeling

- "Language Modeling" is a somewhat confusing label for the task of teaching an algorithm to predict/generate language.
- The standard approach uses the Markov assumption: future words are independent of the past given the present and some finite number of previous rounds.
  - A kth order markov-assumption assumes that the next word in a sequence depends only on the last k words:

$$\Pr(w_{i+1}|w_{1:i}) \approx \Pr(w_{i+1}|w_{i-k:i})$$

- Estimating the probability of a sentence is

$$\Pr(w_{1:n}) \approx \prod_{i=1}^{n} \Pr(w_i|w_{i-k:i-1})$$

where $w_{-k+1}, \ldots, w_0$ are replaced with the padding symbol.

Language Modeling

- "Language Modeling" is a somewhat confusing label for the task of teaching an algorithm to predict/generate language.
- The standard approach uses the Markov assumption: future words are independent of the past given the present and some finite number of previous rounds.
  - A kth order markov-assumption assumes that the next word in a sequence depends only on the last k words:

$$\Pr(w_{i+1}|w_{1:i}) \approx \Pr(w_{i+1}|w_{i-k:i})$$

- Estimating the probability of a sentence is

$$\Pr(w_{1:n}) \approx \prod_{i=1}^{n} \Pr(w_i|w_{i-k:i-1})$$

  where $w_{-k+1}, \ldots, w_0$ are replaced with the padding symbol.
- The task is to learn $\Pr(w_{i+1}|w_{1:i})$ given a large corpus.

Perplexity

- Perplexity is an information-theoretic measurement of how well a probability model predicts a sample.
- Given a text corpus of n words $\{w_1, \ldots w_n\}$ and a language model function $\Pr(\cdot)$, the perplexity is:

$$2^{-\frac{1}{n}\sum_{i=1}^{n} \log \widehat{\Pr}(w_i \mid w_{1:i-1})}$$

- Good language models (i.e., reflective of real language usage) assign high probabilities to the observed words in the corpus, resulting in lower (better) perplexity values.

N-Gram Approach to Language Modeling

- Let $\#(w_{i:j})$ be the count of the sequence of words $w_{i:j}$ in the corpus.
- The MLE estimate for the probability of a word given the previous k words is

$$\widehat{\Pr}(w_{i+1} | w_{i-k:i}) = \frac{\#(w_{i-k:i+1})}{\#(w_{i-k:i})}$$

N-Gram Approach to Language Modeling

- Let $\#(w_{i:j})$ be the count of the sequence of words $w_{i:j}$ in the corpus.
- The MLE estimate for the probability of a word given the previous k words is

$$\widehat{\Pr}(w_{i+1}|w_{i-k:i}) = \frac{\#(w_{i-k:i+1})}{\#(w_{i-k:i})}$$

- The obvious problem:
  - if $w_{i-k:i+1}$ was never observed in the corpus, $\widehat{\Pr}$ is zero, which gives infinite perplexity.
  - zero events are quite common because many phrases are unique.
  - smoothing (adding a small constant to the numerator and denominator) helps.

Neural Language Model Baseline (Goldberg 2017)

- Input:
  - preceding sequence (context words) $w_{1:k}$.
  - V is a finite vocabulary, including special symbols for unknown words, start of sentence, and end of sentence.
  - Each context word is associated with an embedding vector $v(w) \in \mathbb{R}^{n_w}$, the wth row of $\boldsymbol{E}$.
  - The input vector $\boldsymbol{x}$ is a concatenation of the word vectors

Neural Language Model Baseline (Goldberg 2017)

- Input:
  - preceding sequence (context words) $w_{1:k}$.
  - V is a finite vocabulary, including special symbols for unknown words, start of sentence, and end of sentence.
  - Each context word is associated with an embedding vector $v(w) \in \mathbb{R}^{n_w}$, the wth row of $\boldsymbol{E}$.
  - The input vector $\boldsymbol{x}$ is a concatenation of the word vectors
- Output:
  - probability distribution over the next word.

Neural Language Model Baseline (Goldberg 2017)

- Input:
  - preceding sequence (context words) $w_{1:k}$.
  - V is a finite vocabulary, including special symbols for unknown words, start of sentence, and end of sentence.
  - Each context word is associated with an embedding vector $v(w) \in \mathbb{R}^{n_w}$, the wth row of $\boldsymbol{E}$.
  - The input vector $\boldsymbol{x}$ is a concatenation of the word vectors
- Output:
  - probability distribution over the next word.
- The model:

$$
\begin{aligned}
\boldsymbol{x} =& [v(w_1), \ldots, v(w_k)] \\
\boldsymbol{h} =& \boldsymbol{g}(\boldsymbol{x}\boldsymbol{W}_h) \\
\boldsymbol{y} =& \mathrm{softmax}(\boldsymbol{h}\boldsymbol{W}_y)
\end{aligned}
$$

Training Neural Language Models (Goldberg 2017)

$$
\begin{aligned}
\boldsymbol{x} =& [v(w_1), ..., v(w_k)] \\
\boldsymbol{h} =& \boldsymbol{g}(\boldsymbol{x}\boldsymbol{W}_h) \\
\boldsymbol{y} =& \text{softmax}(\boldsymbol{h}\boldsymbol{W}_y)
\end{aligned}
$$

- Training examples are simply each word in the corpus, with the associated k preceding words as the inputs.
- Each word is associated with an n_w-dimensional embedding vector from a row of E, as well as an n_y-dimensional vector from a column of W_y.
  - These are both informative word representations where words that appear in similar contexts will have similar vector representations.
- The computational cost of these language models is the softmax in the final layer, which becomes slower with an increase in vocabulary size.

Outline

Li et al (2016)

- Li et al (2016 learn an embedding vector for each user who wrote a response.
- The intuition:
  - different users have different communication styles, based on their age, gender, social role, background knowledge, personality traits and many other latent factors. By condition-ng on the user when generating the response, the network can learn to adapt its predictions while still using an underlying language model as a backbone.
- As a side effect of training the generator, the network also learns user embeddings, producing similar vectors to users who have similar communication styles. At test time, one can influence the style of the generated response by feeding in a particular user (or average user vector) as a conditioning context.

Outline

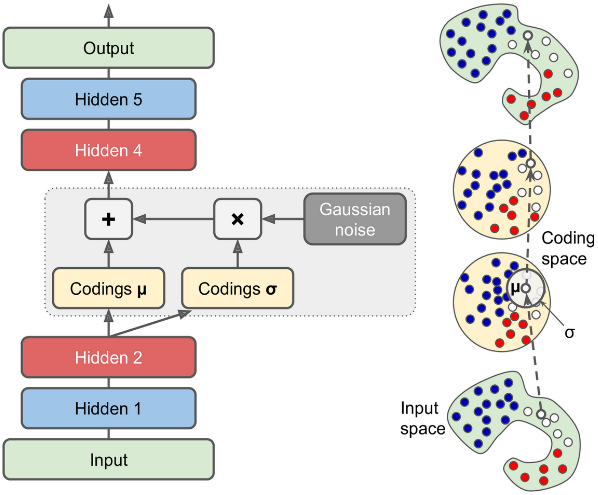
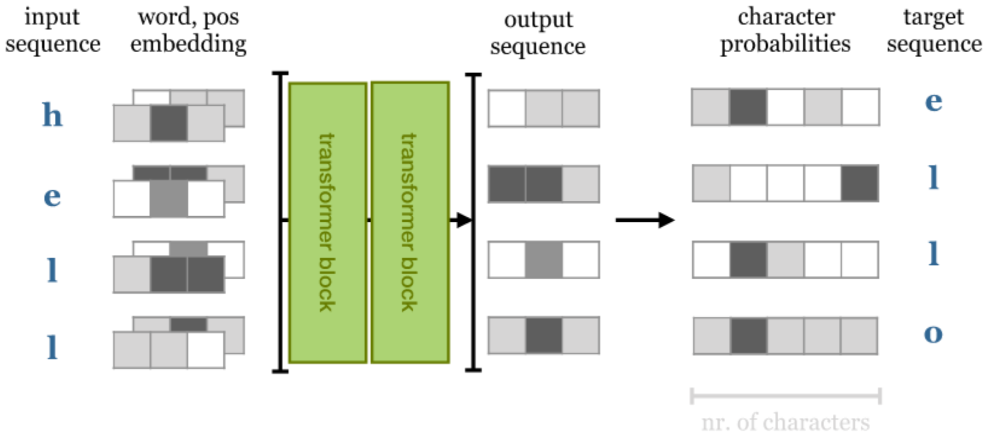


Figure 17-12. Variational autoencoder (left) and an instance going through it (right)

Outline

Text generation transformer



Outline

OPENAI'S NEW MULTITALENTED AI WRITES, TRANSLATES, AND SLANDERS

A step forward in AI text-generation that also spells trouble

By James Vincent | Feb 14, 2019, 12:00pm EST

Howard, co-founder of Fast.AI agrees. "I've been trying to warn people about this for a while," he says. "We have the technology to totally fill Twitter, email, and the web up with reasonable-sounding, context-appropriate prose, which would drown out all other speech and be impossible to filter."

https://transformer.huggingface.co/doc/distil-gpt2

Outline

News Generation Experiment

- ...

Outline

Last Year's Projects (1)

Lazar Peric: GPT Text Generator for Legal Text

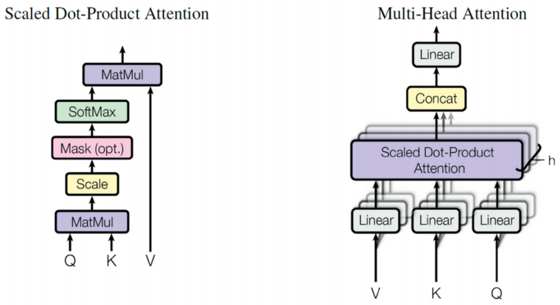


Figure 4.1: Scaled Dot-Product Attention (left) and Multi-Head Attention (right) block. Figure taken from [6]

Last Year's Projects (1)

Lazar Peric: GPT Text Generator for Legal Text

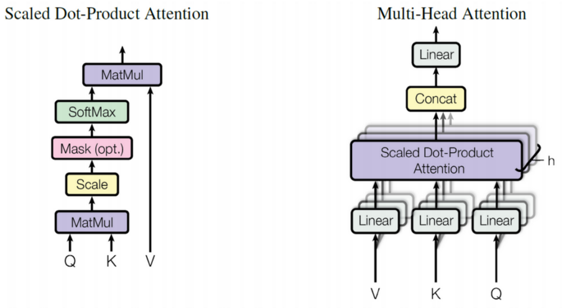


Figure 4.1: Scaled Dot-Product Attention (left) and Multi-Head Attention (right) block. Figure taken from [6]

```
======================================== Start text ========================================
[ * 1209 ] JERRE S. WILLIAMS , Circuit Judge : <eos> This appeal is lodged by H. Roger Lawler
, a debtor in bankruptcy . He claims that an award in bankruptcy of $ <unk> in attorneys
' fees to Richard W. Horton and Vernon O. <unk> is too high . Horton and <unk> are cross -
appealing the amount of the award
====================================== Generated text ======================================
in question , which was based on a $ 1.84 judgment . The district judge found , and the court
concluded that there were sufficient facts in support thereof . We find that there is no
evidence that they are not supported in any way . The judgment is reversed . The judgments
appealed therefrom will stand and will stand and will bear in all other parts of this
judgment , except as they will bear their respective portions of their judgments , together
therewith , with costs of this opinion , with directions that they be reversed and the case
is REMANDED to that portion thereof , and will bear its part with instructions for the new
```

Outline

Generative Adversarial Networks

- The model has two players, generator and discriminator:
  - discriminator says, given an input, what should the label be.
  - generator tries to generate an input that fools the discriminator

Generative Adversarial Networks

- The model has two players, generator and discriminator:
  - discriminator says, given an input, what should the label be.
  - generator tries to generate an input that fools the discriminator
- this has been good for image classification but again, not much in the way of social science applications.

Outline

Reinforcement Learning

- Reinforcement learning has been successful in chess, Go, and other games because the models can play against themselves millions of times.
  - closer to game theory than empirical economics.
  - I have not seen a social-science application of reinforcement learning.