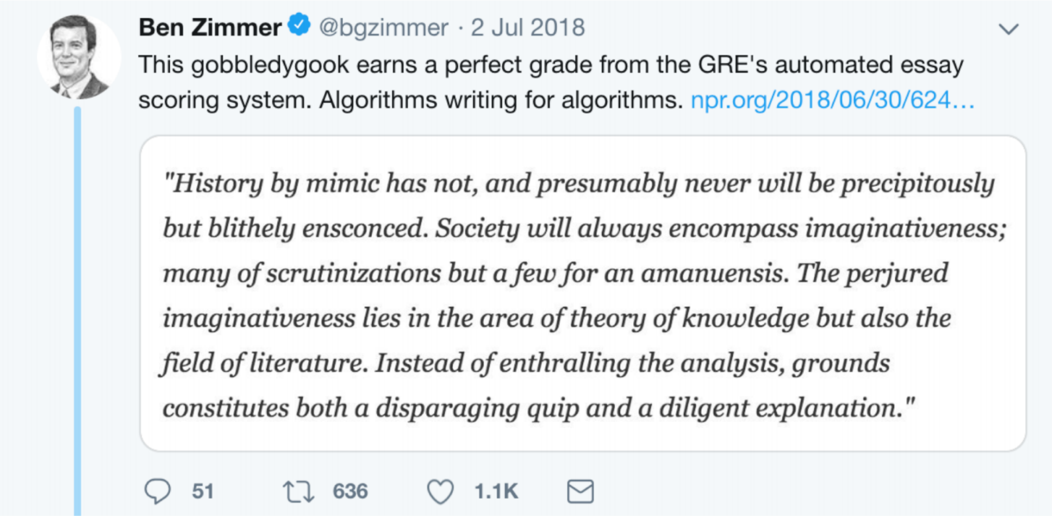
Sequencing Legal DNA

NLP for Law and Political Economy

11. Language Models



Ben Zimmer @bgzimmer · 2 Jul 2018

This gobbledygook earns a perfect grade from the GRE's automated essay scoring system. Algorithms writing for algorithms. npr.org/2018/06/30/624…

"History by mimic has not, and presumably never will be precipitously but blithely ensconced. Society will always encompass imaginativeness; many of scrutinizations but a few for an amanuensis. The perjured imaginativeness lies in the area of theory of knowledge but also the field of literature. Instead of enthralling the analysis, grounds constitutes both a disparaging quip and a diligent explanation."

Outline

Language Modeling

Variational Autoencoders

Text Generation with Transformers

- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL

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 $w_{-k+1}, ..., w_0$ 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 $\{w_1, \dots w_n\}$ and a language model function $\Pr(\cdot)$, the perplexity is:

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

- 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:

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

Training Neural Language Models (Goldberg 2017)

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

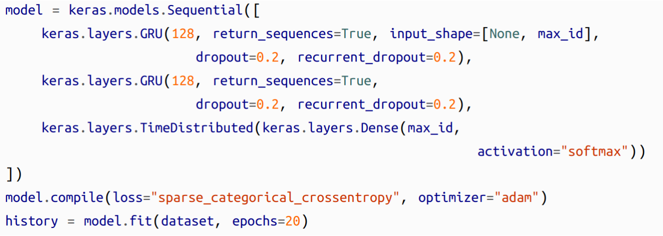
- Training examples are words in the corpus, with the associated k preceding words as the inputs.
- Each word is associated with an n_w-dimensional embedding vector from a row of E, as well as an n_y-dimensional vector from a column of W_y.
  - both could provide good word embeddings.

Training Neural Language Models (Goldberg 2017)

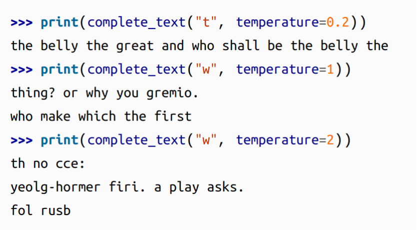
$$\begin{aligned}\boldsymbol{x} =& [v(w_1), \dots, v(w_k)] \\ \boldsymbol{h} =& \boldsymbol{g}(\boldsymbol{x}\boldsymbol{W}_h) \\ \boldsymbol{y} =& \text{softmax}(\boldsymbol{h}\boldsymbol{W}_y)\end{aligned}$$

- Training examples are words 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.
  - both could provide good word embeddings.
- Computational cost of these language models is the softmax in the final layer, which becomes slower with an increase in vocabulary size.

Generating Text with a Keras RNN (Geron Ch. 16)



```
model = keras.models.Sequential([
    keras.layers.GRU(128, return_sequences=True, input_shape=[None, max_id],
                     dropout=0.2, recurrent_dropout=0.2),
    keras.layers.GRU(128, return_sequences=True,
                     dropout=0.2, recurrent_dropout=0.2),
    keras.layers.TimeDistributed(keras.layers.Dense(max_id,
                                                    activation="softmax"))
])
model.compile(loss="sparse_categorical_crossentropy", optimizer="adam")
history = model.fit(dataset, epochs=20)
```



```
>>> print(complete_text("t", temperature=0.2))
the belly the great and who shall be the belly the
>>> print(complete_text("w", temperature=1))
thing? or why you gremio.
who make which the first
>>> print(complete_text("w", temperature=2))
th no cce:
yeolg-hormer firi. a play asks.
fol rusb
```

- Keras has some really useful text pre-processing tools.
- “Temperature” is the level of randomness in the generator.

Beam Search

- Text decoding generally works word by word.
  - but a generated word at any given point might create a low-probability sequence of words.

Beam Search

- Text decoding generally works word by word.
    - but a generated word at any given point might create a low-probability sequence of words.
- Beam search generates multiple words at any given point, and follows those "beams" to generate several branching sequences.
    - after computing the sequences, e.g., 3-4 words, evaluate their probability and only choose beams with relatively high probability.

Conditioned Generation

- Text generators can use metadata, for example on the speaker.
  - e.g., Li et al (2016) learn a categorical embedding for each user who wrote a response, in order to produce automated responses in the style of each user.

Conditioned Generation

- Text generators can use metadata, for example on the speaker.
  - e.g., Li et al (2016) learn a categorical embedding for each user who wrote a response, in order to produce automated responses in the style of each user.
- As a side effect of training the generator, the network 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

Language Modeling

Variational Autoencoders

Text Generation with Transformers

- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL

Autoencoders can memorize complex data

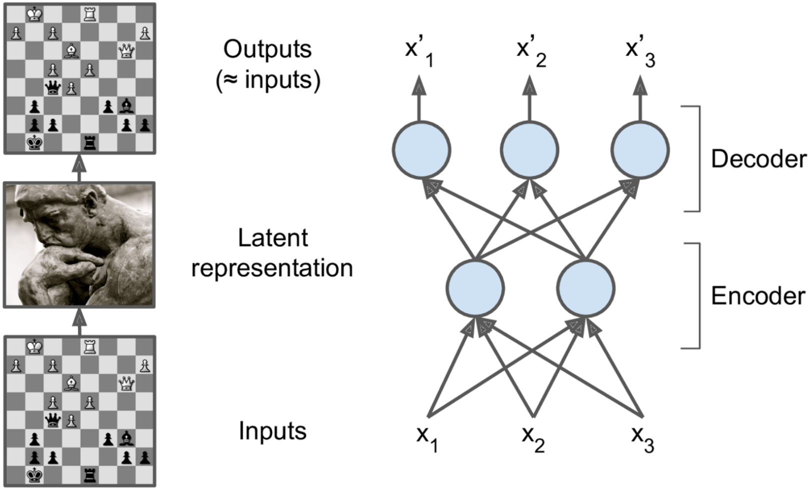


Figure 17-1. The chess memory experiment (left) and a simple autoencoder (right)

- can they memorize low-dimensional encodings and then reproduce text?

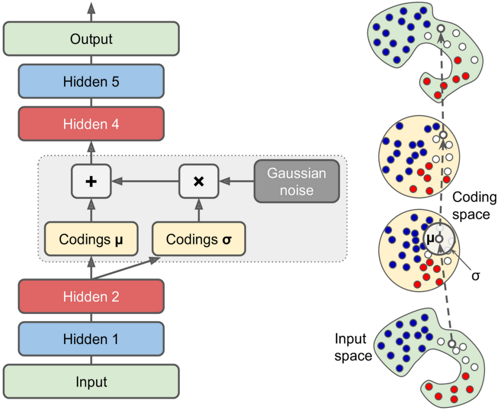


Figure 17-12. Variational autoencoder (left) and an instance going through it (right)

- Variational Autoencoder transforms low-dimensional encodings to the parameters of a gaussian (mean μ and variances σ^2), then draws from the distribution to produce first layer for the decoder.

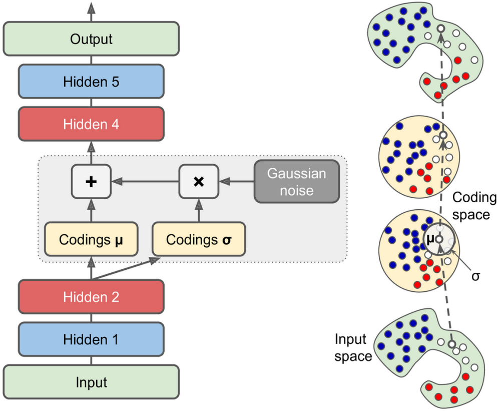


Figure 17-12. Variational autoencoder (left) and an instance going through it (right)

- Can then sample from the normal distribution (or just choose numbers) and generate reconstructions.

- Variational Autoencoder transforms low-dimensional encodings to the parameters of a gaussian (mean μ and variances σ^2), then draws from the distribution to produce first layer for the decoder.

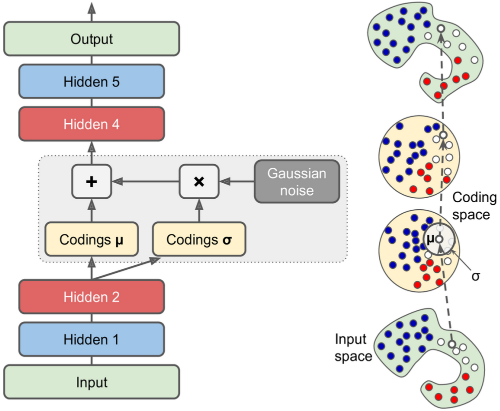
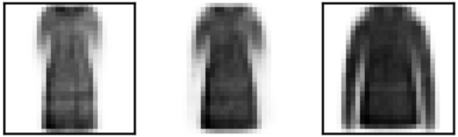


Figure 17-12. Variational autoencoder (left) and an instance going through it (right)

- Variational Autoencoder transforms low-dimensional encodings to the parameters of a gaussian (mean μ and variances σ^2), then draws from the distribution to produce first layer for the decoder.

- Can then sample from the normal distribution (or just choose numbers) and generate reconstructions.
- The amazing thing about VAE's is *semantic interpolation*:
  picking an encoding vector between two encodings will produce a reconstruction that is in between the associated images/documents:



- doesn't seem to work will with text (but a nice replication exercise).

Outline

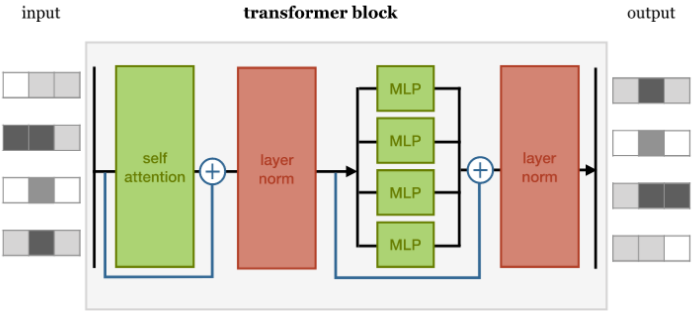
Language Modeling

Variational Autoencoders

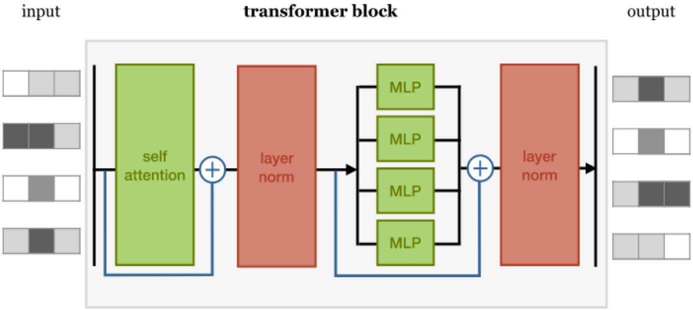
Text Generation with Transformers

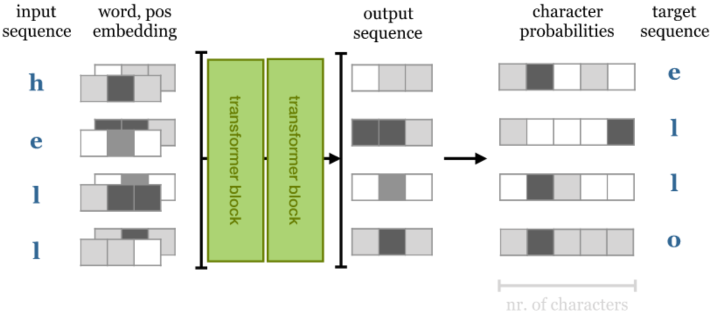
- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL

Text generation transformer

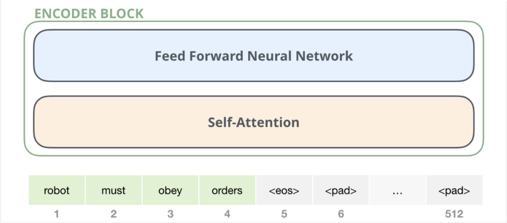


Text generation transformer

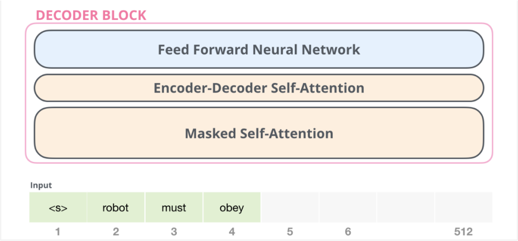




Encoder Blocks vs. Decoder Blocks

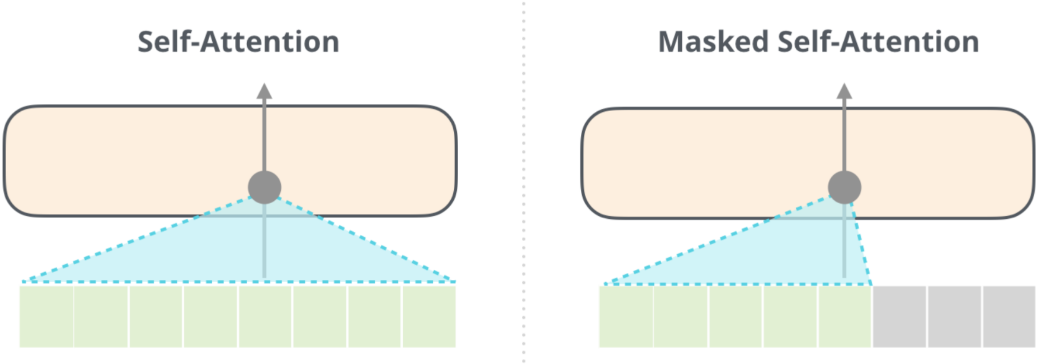


An encoder block from the original transformer paper can take inputs up until a certain max sequence length (e.g. 512 tokens). It's okay if an input sequence is shorter than this limit, we can just pad the rest of the sequence.



- Unlike the encoder block (which observes the whole sequence), the decoder block masks future tokens.
  - Autoregressive models (like GPT2) use a special layer architecture ("masked self-attention") that blocks information on right side of the sequence.
  - this is different from bidirectional models (like BERT), which only have encoder blocks but substitute the [mask] token.

Masked Self-Attention



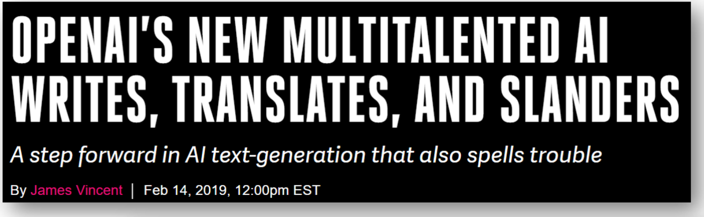
Outline

Language Modeling

Variational Autoencoders

Text Generation with Transformers

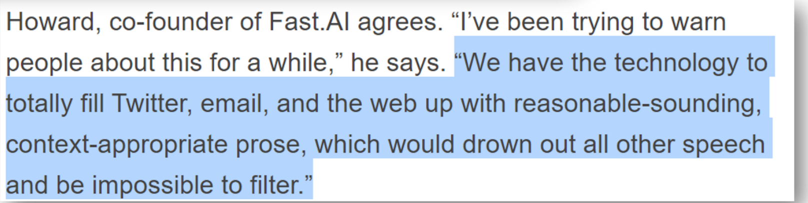
- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL



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`

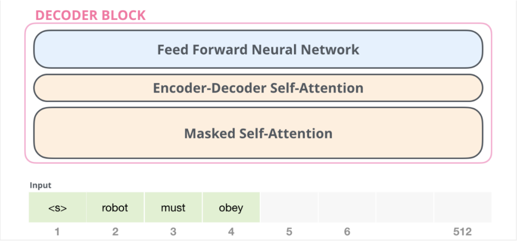
Corpus and Vocabulary

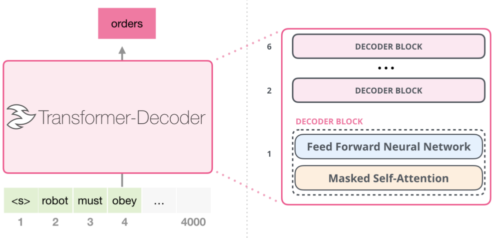
- GPT-2 was trained on all articles linked from Reddit posts with at least three net up-votes.
  - 8 million documents
  - 40 GB of text

Corpus and Vocabulary

- GPT-2 was trained on all articles linked from Reddit posts with at least three net up-votes.
  - 8 million documents
  - 40 GB of text
- Subword tokenization with byte-pair encoding (similar to SentencePiece)
  - vocabulary = 50,000 word segments
  - small model: 768-dimensional word embeddings
  - big model: 1600-dimensional word embeddings

GPT-2 Architecture





The decoder blocks are identical. I have expanded the first one so you can see its self-attention layer is the masked variant. Notice that the model now can address up to 4,000 tokens in a certain segment -- a massive upgrade from the 512 in the original transformer.

- GPT-2 encodes sequences up to 1024 (sub-word) tokens in length.
- The largest model has 48 stacked decoder block layers and 1.5B learnable parameters.

What can GPT-2 Do?

"Language models are unsupervised multitask learners"

- State-of-the-art perplexity on diverse corpora.

What can GPT-2 Do?

"Language models are unsupervised multitask learners"

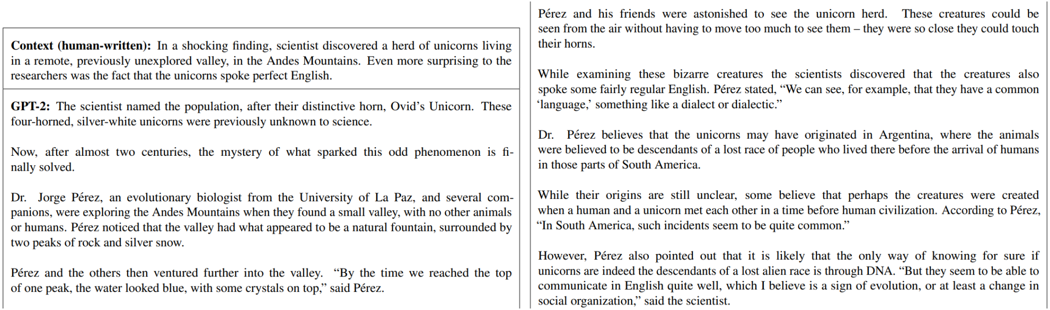
- State-of-the-art perplexity on diverse corpora.
- Reading Comprehension: *`<context> <question> A:`*
- Summarization: `TL;DR:`
- Question Answering: `A:`

What can GPT-2 Do?

“Language models are unsupervised multitask learners”

- State-of-the-art perplexity on diverse corpora.
- Reading Comprehension: *<context> <question> A:*
- Summarization: `TL;DR:`
- Question Answering: `A:`
- Translation:
  - *[English sentence 1] = <French sentence 1>*
  - *[English sentence 2] = <French sentence 2>*
  - *.....*
  - *[Source sentence] =*

Story Generation



Context (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.

GPT-2: 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."

Dr. Pérez believes that the unicorns may have originated in Argentina, where the animals were believed to be descendants of a lost race of people who lived there before the arrival of humans in those parts of South America.

While their origins are still unclear, some believe that perhaps the creatures were created when a human and a unicorn met each other in a time before human civilization. According to Pérez, "In South America, such incidents seem to be quite common."

However, Pérez also pointed out that it is likely that the only way of knowing for sure if unicorns are indeed the descendants of a lost alien race is through DNA. "But they seem to be able to communicate in English quite well, which I believe is a sign of evolution, or at least a change in social organization," said the scientist.

BERT vs GPT2

- GPT2 is autoregressive, allowing it to generate text.
- BERT is bidirectional, so it cannot do text generation.
  - But uses context from both sides of a word to get better word prediction accuracy.

Outline

Language Modeling

Variational Autoencoders

Text Generation with Transformers

- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL

Grover

- Grover is a modified version of GPT-2.
- Goal: generate fake newspaper articles

Grover

- Grover is a modified version of GPT-2.
- Goal: generate fake newspaper articles
- next predicted word is based not only on previous words (text body), but also on metadata:

$$p(\text{domain}, \text{date}, \text{authors}, \text{headline}, \text{body}).$$

  - newspaper domain of article
  - publication date
  - author names
  - article headline
  - text body

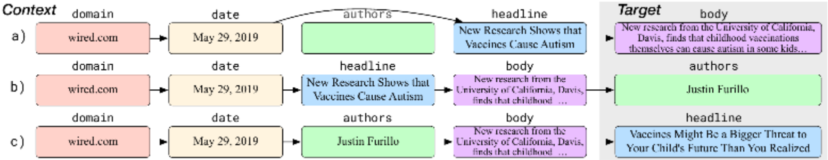

Context
Target
a)
domain
wired.com
date
May 29, 2019
authors
headline
New Research Shows that Vaccines Cause Autism
body
New research from the University of California, Davis, finds that childhood vaccinations themselves can cause autism in some kids...
b)
domain
wired.com
date
May 29, 2019
headline
New Research Shows that Vaccines Cause Autism
body
New research from the University of California, Davis, finds that childhood ...
authors
Justin Furillo
c)
domain
wired.com
date
May 29, 2019
authors
Justin Furillo
body
New research from the University of California, Davis, finds that childhood ...
headline
Vaccines Might Be a Bigger Threat to Your Child's Future Than You Realized

Figure 2: A diagram of three GROVER examples for article generation. In row a), the body is generated from partial context (the authors field is missing). In b), the model generates the authors. In c), the model uses the new generations to regenerate the provided headline to one that is more realistic.

- in this example, user specifies domain, date, and headline. Grover generates a) text body, then b) authors, then c) more realistic headline.

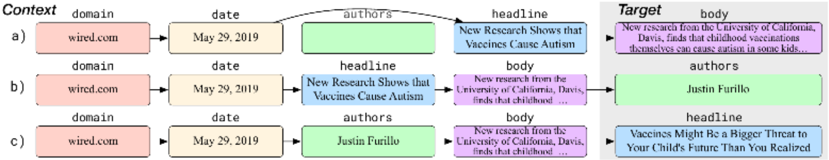

Context
Target
a)
domain
wired.com
date
May 29, 2019
authors
headline
New Research Shows that Vaccines Cause Autism
body
New research from the University of California, Davis, finds that childhood vaccinations themselves can cause autism in some kids...
b)
domain
wired.com
date
May 29, 2019
headline
New Research Shows that Vaccines Cause Autism
body
New research from the University of California, Davis, finds that childhood ...
authors
Justin Furillo
c)
domain
wired.com
date
May 29, 2019
authors
Justin Furillo
body
New research from the University of California, Davis, finds that childhood ...
headline
Vaccines Might Be a Bigger Threat to Your Child's Future Than You Realized

Figure 2: A diagram of three Grover examples for article generation. In row a), the body is generated from partial context (the authors field is missing). In b), the model generates the authors. In c), the model uses the new generations to regenerate the provided headline to one that is more realistic.

- in this example, user specifies domain, date, and headline. Grover generates a) text body, then b) authors, then c) more realistic headline.
- fields are dropped with probability 10%, and all but body with probabiltiy 35% → Grover learns to perform unconditional generation.

RealNews Corpus

- Authors scraped all news articles from the 5000 news domains from Google News.
- Parsed using Newspaper Python library
- 120 GB of uncompressed text after deduplication
- Training Set: December 2016 - March 2019
- Test Set: April 2019

RealNews Corpus

- Authors scraped all news articles from the 5000 news domains from Google News.
- Parsed using Newspaper Python library
- 120 GB of uncompressed text after deduplication
- Training Set: December 2016 - March 2019
- Test Set: April 2019

- instances are randomly samples 1024-token sequences.

Results

- Grover can generate articles in particular newspaper styles.
  - metadata is important: it reduced perplexity from 9.3 to 8.7

Results

- Grover can generate articles in particular newspaper styles.
  - metadata is important: it reduced perplexity from 9.3 to 8.7
- Grover is better than human generated fake news:

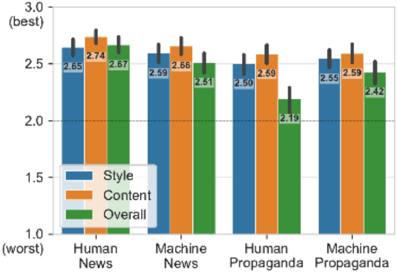


Figure 4: Human evaluation. For each article, three annotators evaluated style, content, and the overall trustworthiness; 100 articles of each category were used. The results show that propaganda generated by Grover is rated more plausible than the original human-written propaganda.

`https://grover.allenai.org/`

Fake News Detection

- They take Grover's document embedding for the true article and generated articles, and feed that to a true/false classifier.
    - they do it in the test set month (April 2019); training ended in March 2019.

Fake News Detection

- They take Grover's document embedding for the true article and generated articles, and feed that to a true/false classifier.
  - they do it in the test set month (April 2019); training ended in March 2019.
- Grover can identify its own fake articles with 81% accuracy.
  - better than other models including BERT

Outline

Language Modeling

Variational Autoencoders

Text Generation with Transformers

- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL

Plug and Play Language Model (PPLM)

- PPLM combines a large (pre-trained) language model with a small auxiliary attribute model for conditioned text generation.

PPLM: Theory

- Setup:
  - Let $p(w) =$ unconditional language model (LM), a probability distribution for word w given history w_0

PPLM: Theory

- Setup:
  - Let $p(w)$ = unconditional language model (LM), a probability distribution for word w given history w_0
  - $p(w|a)$ = conditional LM, generating sentences with attribute a.

PPLM: Theory

- Setup:
  - Let $p(w)$ = unconditional language model (LM), a probability distribution for word w given history w_0
  - $p(w|a)$ = conditional LM, generating sentences with attribute a.
- By Bayes rule:

$$p(w|a) \propto p(a|w)p(w)$$

  - can be approximated efficiently using the method in Ngyuen et al (2016).

PPLM: Theory

- Setup:
  - Let $p(w)$ = unconditional language model (LM), a probability distribution for word w given history w_0
  - $p(w|a)$ = conditional LM, generating sentences with attribute a.
- By Bayes rule:

$$p(w|a) \propto p(a|w)p(w)$$

  - can be approximated efficiently using the method in Ngyuen et al (2016).
- PPLM insight:
  - instead of training $p(w|a)$ from scratch, take pre-trained $p(w)$, learn auxiliary model $p(a|w)$, and approximate $p(w|a)$ using Bayes rule.

PPLM: Approach

At each word generation step, we have history w_0 represented as the LM's hidden state vector h.

PPLM: Approach

At each word generation step, we have history w_0 represented as the LM's hidden state vector h.

1. compute

$$\begin{aligned} \log p(w) \\ \log p(a|w) \\ \nabla_h \log p(w) \\ \nabla_h \log p(a|w) \end{aligned}$$

- these gradients are available using an efficient forward and backward pass of both models.

PPLM: Approach

At each word generation step, we have history w_0 represented as the LM's hidden state vector h.

1. compute

$$\begin{aligned} \log p(w) \\ \log p(a|w) \\ \nabla_h \log p(w) \\ \nabla_h \log p(a|w) \end{aligned}$$

    - these gradients are available using an efficient forward and backward pass of both models.

2. Use the gradients to move h a small step (α) in the direction of increasing $\log(p(a|x))$ and increasing $\log(p(x))$.
3. Sample the next word using the shifted state h.

PPLM: Approach

At each word generation step, we have history w_0 represented as the LM's hidden state vector h.

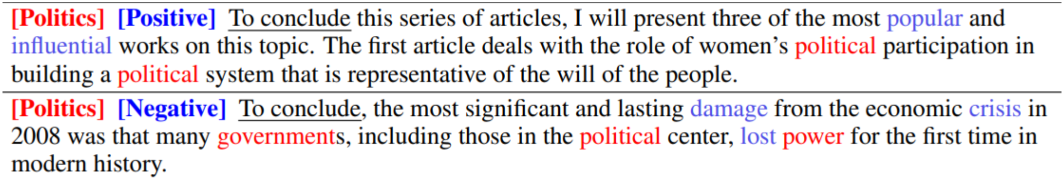
1. compute

$$\begin{aligned} \log p(w) \\ \log p(a|w) \\ \nabla_h \log p(w) \\ \nabla_h \log p(a|w) \end{aligned}$$

   - these gradients are available using an efficient forward and backward pass of both models.

2. Use the gradients to move h a small step (α) in the direction of increasing $\log(p(a|x))$ and increasing $\log(p(x))$.
3. Sample the next word using the shifted state h.

→ Moving h to increase both the attribute model *and* the language model works to maintain fluency of generated language.



[Politics] **[Positive]** To conclude this series of articles, I will present three of the most popular and influential works on this topic. The first article deals with the role of women's political participation in building a political system that is representative of the will of the people.

[Politics] **[Negative]** To conclude, the most significant and lasting damage from the economic crisis in 2008 was that many governments, including those in the political center, lost power for the first time in modern history.

Outline

Language Modeling

Variational Autoencoders

Text Generation with Transformers

- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL

Kreps et al: News Generation Experiment

- Kreps et al (2019) evaluate the use of GPT-2 for fake news generation.
- Experiment:
  - New York Times story on North Korea.
  - GPT-2 gets 2 sentences, then generates 20 short news stories.
  - Researchers manually selected the most credible out of these twenty.

Kreps et al: News Generation Experiment

- Kreps et al (2019) evaluate the use of GPT-2 for fake news generation.
- Experiment:
  - New York Times story on North Korea.
  - GPT-2 gets 2 sentences, then generates 20 short news stories.
  - Researchers manually selected the most credible out of these twenty.
  - Respondents rank stories as credible or not.
- Results:
  - For the larger GPT-2 models, machine-generated articles were rated the same as the true article.

Peng et al: Generating Normative Text

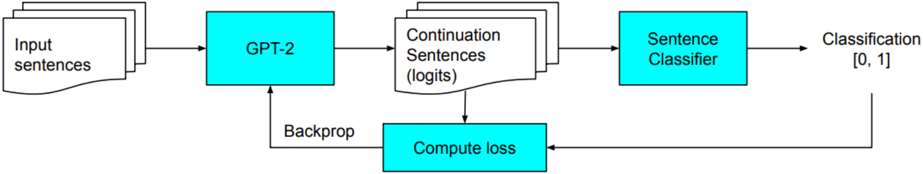


Figure 1: Pipeline for fine-tuning GPT-2 with a classifier.

Peng et al: Generating Normative Text

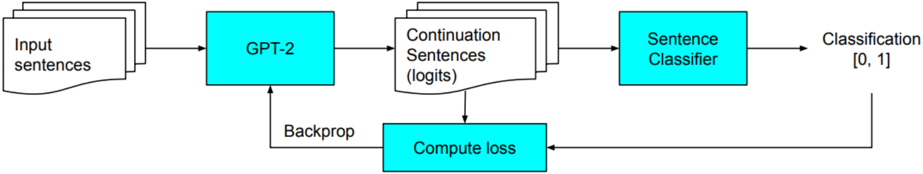
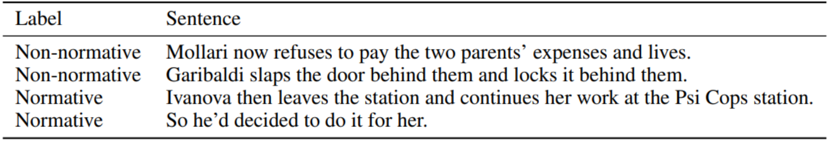


Figure 1: Pipeline for fine-tuning GPT-2 with a classifier.



Label	Sentence
Non-normative	Mollari now refuses to pay the two parents' expenses and lives.
Non-normative	Garibaldi slaps the door behind them and locks it behind them.
Normative	Ivanova then leaves the station and continues her work at the Psi Cops station.
Normative	So he'd decided to do it for her.

Table 1: Examples of generated sentences from the model trained with the normative classifier.

Outline

Language Modeling

Variational Autoencoders

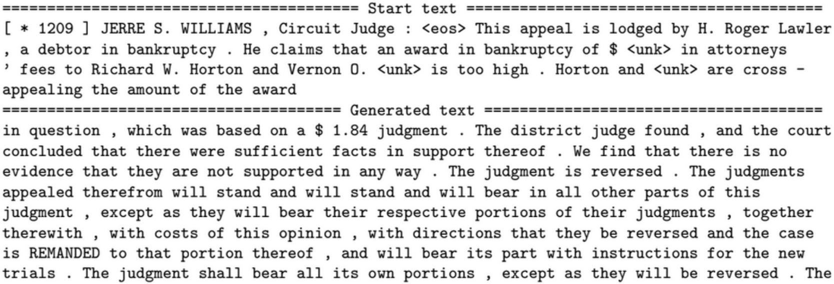
Text Generation with Transformers

- GPT-2
- Grover: Conditioning on News Article Metadata
- PPLM: Conditioning with Auxiliary Attribute Model
- Other Applications
- Transformer XL

Relative Positional Embedding

- Position vectors encode not the absolute position, but the distance to the current output.
  - This requires moving the position encoding into the attention mechanism.
- The resulting transformer generalizes better to sequences of unseen length.

Ash, Mijic, and Peric (2020)

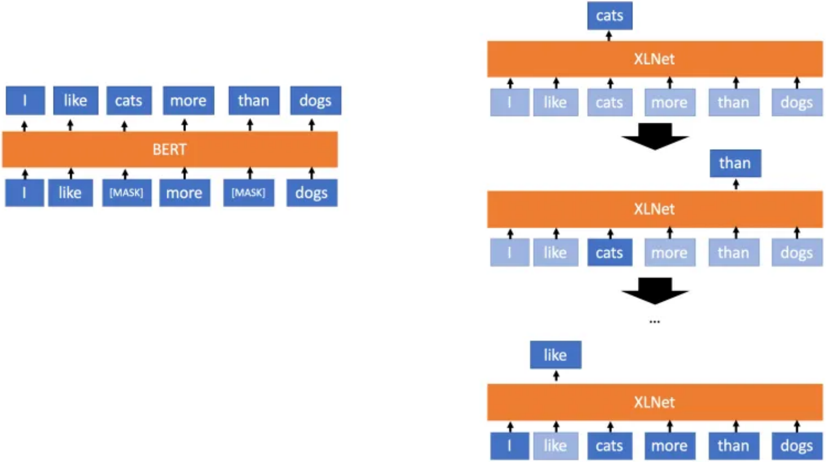


```
======================================== Start text ========================================
[ * 1209 ] JERRE S. WILLIAMS , Circuit Judge : <eos> This appeal is lodged by H. Roger Lawler
, a debtor in bankruptcy . He claims that an award in bankruptcy of $ <unk> in attorneys
' fees to Richard W. 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
trials . The judgment shall bear all its own portions , except as they will be reversed . The
```

- would be great to add the conditioned generation for this task, like Grover (date, judge name, party names, topics, etc)

XLNet = BERT + Transformer XL

- replace BERT's sequential masking task (predict each word in order) with permutation language modeling (predict masked words in random order).



- uses relative positional embeddings (from Transformer XL) so that it can accept sequences of any length.