Neural Network Language Models

Natural Language Processing

Table of Contents

Motivation and Foundations

N-gram LMs

- Language models (LMs) calculate the probability of the next word in a sequence given the preceding ones
- One popular LM implementation are the so-called N-gram LMs
- N-gram LMs very successful in many NLP tasks
- N-gram LMs define a probability distribution for each word given the preceding ones:

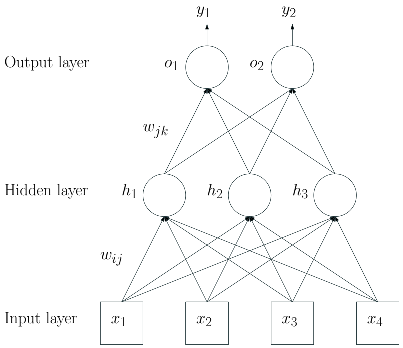
$$p(x|h) = \frac{C(h, x)}{C(h)}$$

Limitations of N-gram LMs

- Many histories are similar but exact match of h is assumed
- Arbitrary representation of words
- Poor at handling uncommon or unseen events
- Poor at capturing long term relationships between words
- Curse of dimensionality

Neural Networks

- Neural networks (NNs) are statistical learning algorithms that estimate functions from a set of inputs
- NNs composed of connected neurons usually grouped in layers
- The canonical example is the feedforward NN:



Neural Networks

- Neurons are processing units computing weighted sums of their inputs
    - Each connection has an associated weight
    - Output is given by an activation function over the weighted sum
- Three parameters define a specific NN:
    - Interconnection pattern between the neurons
    - Learning process used to update or train the weights
    - Activation function converting the neuron input into its output

Neural Network Language Models

- NNs can be used to build LMs, removing or alleviating the limitations of N-gram LMs
- The history h is projected into some continuous low-dimensional space, where similar histories get clustered
- Thanks to parameter sharing among similar histories, the model is more robust: less parameters have to be estimated

Deep Learning

- Early NN-based algorithms such as the multilayer perceptron were typically composed of at most 3 layers
- Each layer can be seen as a mathematical representation of the input
- Multiple layers allow to capture the different factors or features that are relevant for a particular learning task
    - Opposed to hand-made features

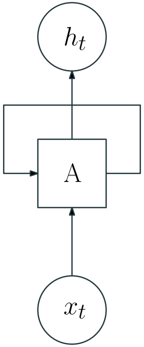
Deep Learning

- Deep learning models are NNs composed of many layers
- Deep learning has been made possible due to some advances:
    - Availability of large amounts of training data
    - More powerful computer infrastructures
    - Better optimization algorithms
- State-of-the-art LMs follow a deep architecture: transformer models

Recurrent Models

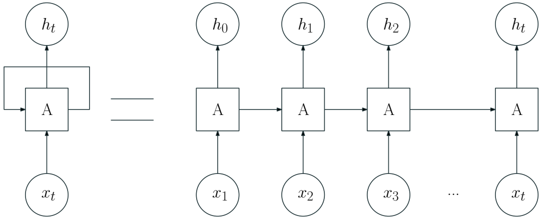
Recurrent Neural Networks

- RNNs are just networks with loops
- Loops allow the algorithm to capture long-range dependencies



Recurrent Neural Networks

- It's typical to represent RNNs unrolled in time:



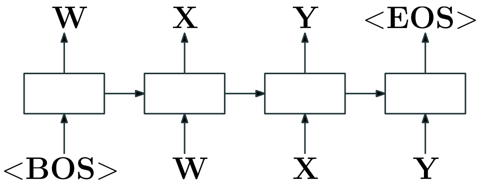
- In simple RNNs input is processed in one direction
- Bidirectional RNNs concatenates two simple RNNs in both directions

Specializations of the basic RNN

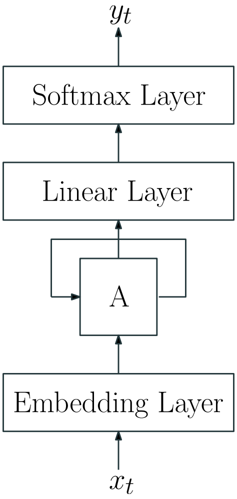
- Initial implementations of the recurrent unit (A) were very simple, summing 2 matrix multiplications and applying the activation function
- This configuration was not able to preserve information over many timesteps due to the vanishing gradients problem
- To avoid this, alternative recurrent units have been defined:
    - **GRU** (Gated Recurrent Unit): use reset and update gates to determine the information to be retained for future predictions (Cho et al. 2014)
    - **LSTM** (Long Short Term Memory): similar to GRU units but using three gates instead of two (Hochreiter and Schmidhuber 1997)

RNN Language Models

- RNNs can be used to implement LMs (Mikolov et al. 2010)



RNN Language Models: Architecture



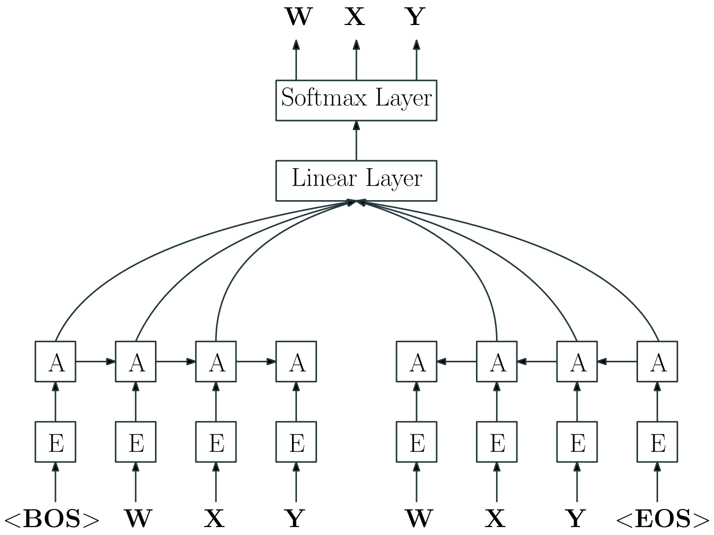
RNN Language Models: Architecture

- **Softmax layer**: converts output to probability distribution
- **Linear layer**: maps RNN output to an alphabet size vector
- **Embedding layer**: maps categorical variable to a continuous one

Bidirectional RNN Language Models

- RNN LMs only take into account left context when predicting next symbol
- Bidirectional RNN LMs enables use of both right and left contexts

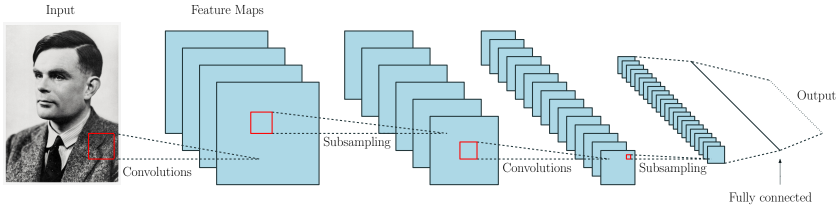
Bidirectional RNN Language Models



Convolutional Models

Convolutional Neural Networks for Image Analysis

- Convolutional neural networks (CNNs) are a class of NNs originally used for image analysis
- Biologically-inspired connection pattern (visual cortex)



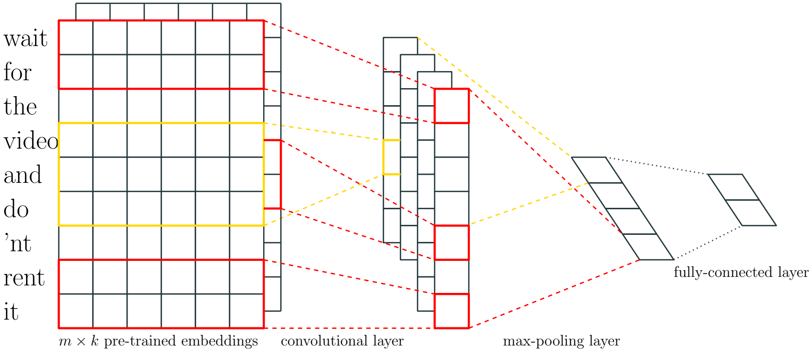
Convolutional Neural Networks for Image Analysis

- Convolutions are learned filters that identify features in the input
- A limitation of this approach is that it records the precise position of the features in the input
- One way to alleviate this problem is subsampling, where a lower resolution version of the input is created
- Subsampling is typically implemented by means of a pooling layer

Convolutional Neural Networks for Text Classification

- CNNs can also be applied to NLP, although initially they were not used for language modelling
- One important application of CNNs in NLP would be text classification
- Seminal work applied convolution and pooling operations (Kim 2014)

Convolutional Neural Networks for Text Classification

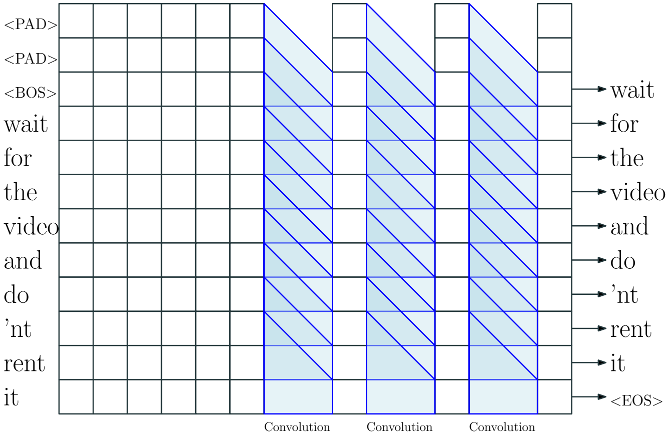


Convolutional Neural Networks for Language Modeling

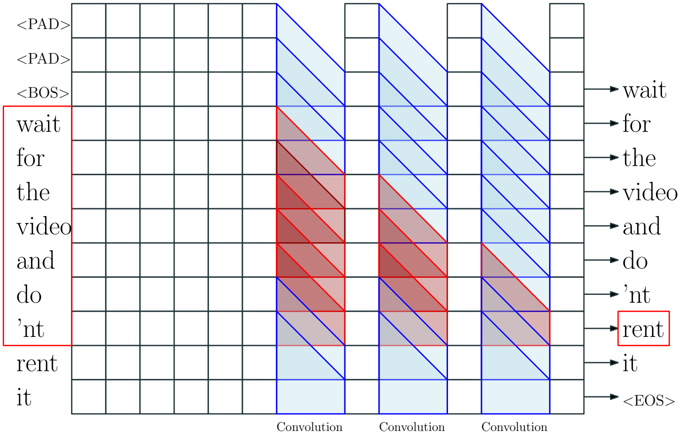
- CNNs for language modeling require some adaptations:
    - Pooling is not executed so as to keep positional information
    - To predict a new word it's necessary to ensure that only the previous ones are considered (padding symbols are used when necessary)
    - Context size is fixed, but it can be very long when stacking many layers
    - Stacking many layers make training difficult, requiring the use of specific techniques such as residual connections

- See for example (Dauphin et al. 2016) for more information

Convolutional Neural Networks for Language Modeling



Convolutional Neural Networks for Language Modeling



Sequence to Sequence Models

Conditional Language Models

- Conventional language models estimate the probability $p(y)$ of a sequence of tokens $(y_1, y_2, ..., y_n)$
- In conditional language models, the unconditional probability $p(y)$ is replaced by a conditional one:

$$p(y|x) = p(y_1, y_2, ..., y_n|x)$$

- x can be defined in multiple ways (text, image, acustic signal)
- Sequence-to-sequence (Seq-to-Seq) models estimate $p(y|x)$

Machine Translation

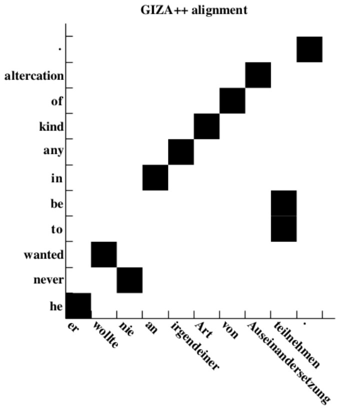
- Seq-to-Seq models (Sutskever et al. 2014) were initially defined to solve machine translation (MT) tasks
- For a given input sentence x, the statistical approach to MT finds the translation of highest probability in the output language, y

$$\hat{y} = \arg\max_{y}\{P(y|x)\} = \arg\max_{y}\{P(y) \cdot P(x|y)\}$$

- Original statistical MT systems modeled the distributions after applying the Bayes rule:
  - $P(y)$ modeled with an N-gram language model
  - $P(x|y)$ modeled with an HMM-based alignment model

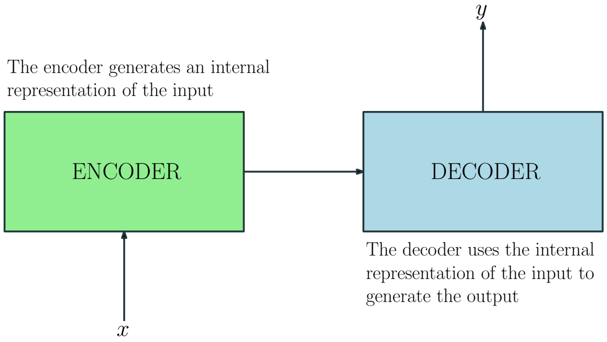
Machine Translation: Alignments

- Input to output alignment is a key idea in MT
- It can be graphically represented as a matrix
- HMM-based MT technology generates alignments as a by-product



The Encoder-Decoder Framework

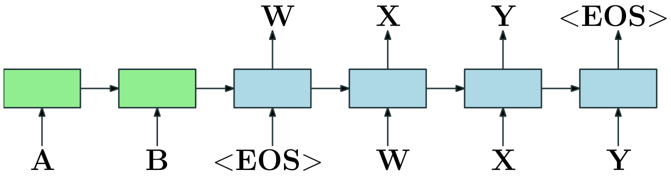
- The encoder-decoder framework is the standard modeling paradigm for Seq-to-Seq tasks



y
The encoder generates an internal representation of the input
ENCODER
DECODER
The decoder uses the internal representation of the input to generate the output
x

Seq-to-Seq Models

- Original Seq-to-Seq model concatenates two RNN LMs (Sutskever et al. 2014)
- First RNN LM encodes input sequence into a vector
- Second RNN LM decodes this vector into the output sequence

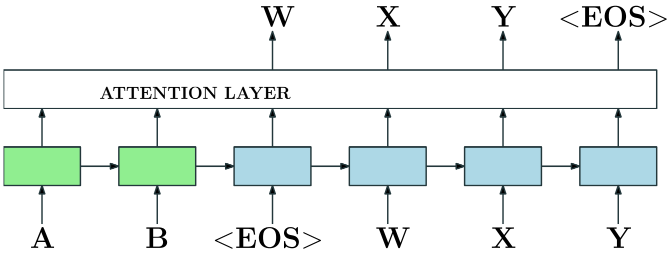


Seq-to-Seq Plus Attention Models

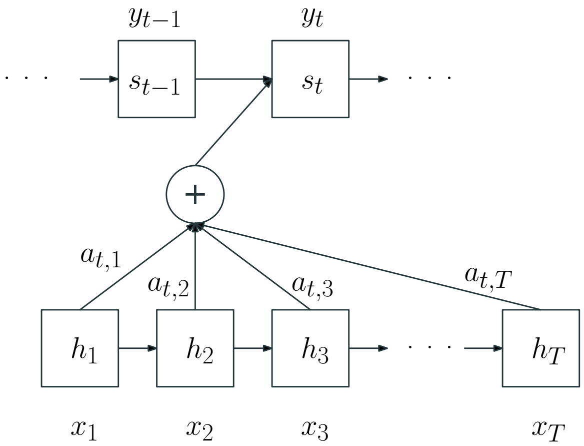
- Original Seq-to-Seq models had problems to capture long range dependencies between symbols
  - the signal should propagate through many time steps in the encoder
  - the decoder should work with just one representation of the input
  - difficult to reflect that each output token can be related to different parts of the input
- The solution is the attention mechanism (Bahdanau et al. 2015)
  - when generating a new output token, the decoder is granted access to all of the encoder states
  - this capability is used to decide which input parts are more relevant

Seq-to-Seq Attention Mechanism

- The Seq-to-Seq architecture is modified to include an attention layer

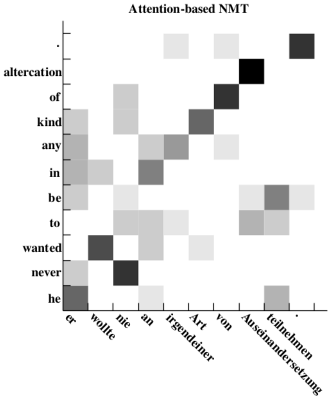


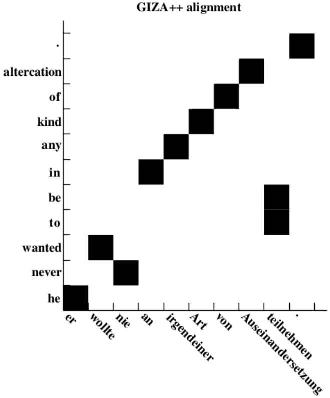
Seq-to-Seq Attention Mechanism (Detailed)



Seq-to-Seq Attention Mechanism

- The attention mechanism computes the relevance of the encoder states for each output token as a distribution
- If this information is represented graphically, we get something similar to the MT alignments we've seen before





Transformer-based Models

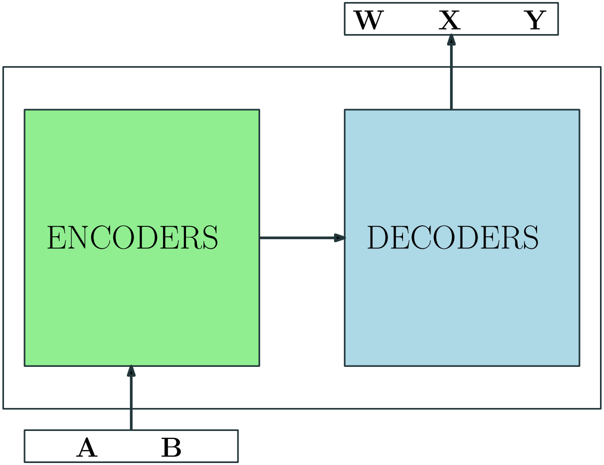
Introduction

- RNN training is difficult to parallelize
- CNNs fast to train but not naturally designed to handle symbol dependencies
- How could we define a model that combines the best of both worlds?

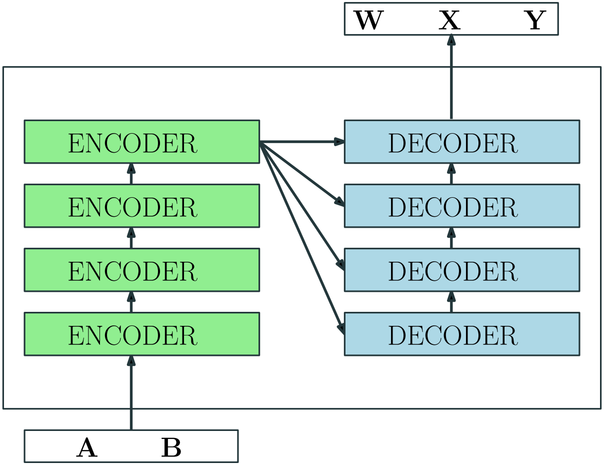
Transformer Models

- Transformer models (Vaswani et al. 2017) are based solely on attention mechanisms (no recurrence or convolutions)
- Its definition enables huge training speed gains with respect to RNNs
- Transformer models constitute the state-of-the-art in language and Seq-to-Seq modelling
- Transformer models also adopt the encoder-decoder framework

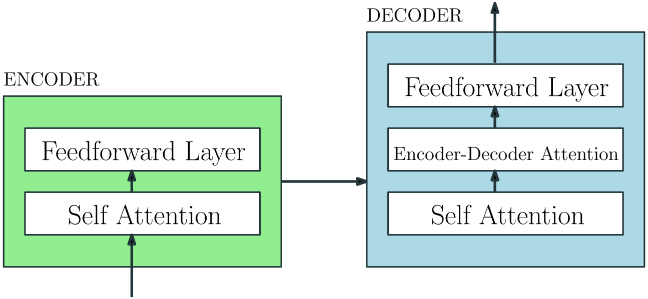
Transformer Models



Transformer Models



Transformer Models



Transformer Models

- **Encoder Self Attention**: all encoder states are received at one and this module establishes relationships between them
- **Decoder Self Attention**: basically the same as encoder attention but the output tokens are generated one at a time. During training masking is used to forbid the decoder to look ahead
- **Encoder-Decoder Attention**: this module establish relationships between encoder and decoder states

Transformer Models and Artificial Intelligence Explainability

- Explainability of NN model's output is an open research problem
- The attention mechanism can be useful for this purpose
- RNN Seq-to-Seq attention mechanism produces one attention matrix
- Transformer models produce multiple attention matrices
- There are works that try to interpret this information (Voita et al. 2019)

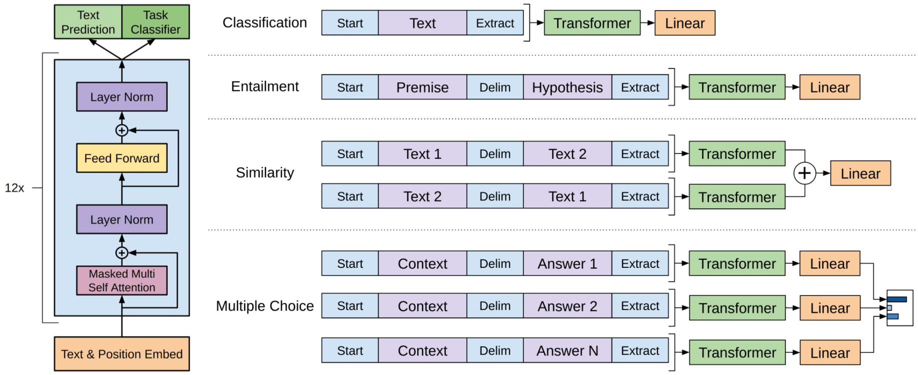
Transfer Learning

- The data required to train a model for a particular task is often scarce
- However, sometimes there is plenty of data for a related task
- Transfer learning techniques try to transfer knowledge from one task to another
- Most relevant technique for this presentation is *model pre-training*
    - a model trained for a general task is *fine-tuned* for a more specific or *downstream task*
    - used in two popular models: GPT and BERT

Generative Pre-Training for Language Understanding

- Generative Pre-Training (GPT) (Radford et al. 2018) is a left-to-right language model
- GPT uses a 12-layer transformer decoder with no decoder-encoder attention
- At the pre-training stage, the model is trained to predict the next token of a sequence
- At the fine-tuning stage, the model allows to perform more specific tasks (e.g. sentence classification)

Generative Pre-Training for Language Understanding

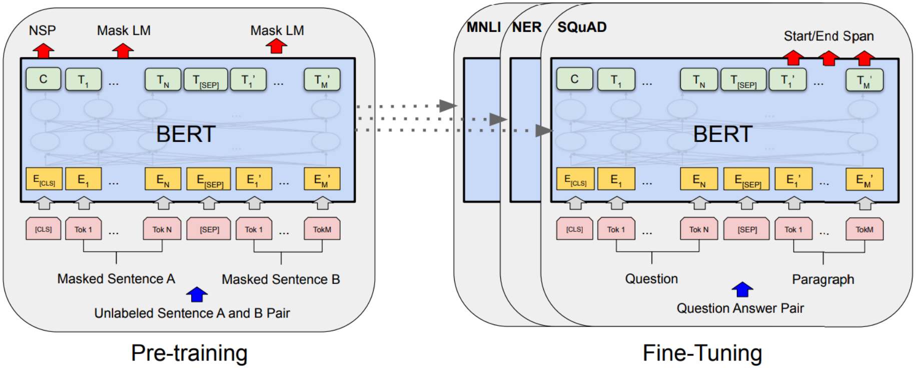


NOTE: Figure taken from GPT's original paper

Bidirectional Encoder Representations from Transformers

- Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. 2019) is a bidirectional language model
- BERT is a transformer's encoder
- At the pre-training stage, BERT learns two tasks:
  - predict masked tokens in a sequence
  - determine whether a sequence of tokens goes after another one
- At the fine-tuning stage, the model can be specialized to carry out tasks similar to those solved by GPT

Bidirectional Encoder Representations from Transformers



NOTE: Figure taken from BERT's original paper

The Era of Foundation Models

Foundation Models

- GPT and BERT are examples of an emerging paradigm in artificial intelligence (AI): the so-called *foundation models* (Bommasani et al. 2021)
- A foundation model is a model trained on broad data that can be adapted to a wide range of downstream tasks
- Foundation models are based on deep learning and transfer learning
- Foundation models are closely linked to the concepts of *homogeneization* and *emergence*

Foundation Models: Homogeneization

- The introduction of BERT marks the beginning of the era of foundation models, characterized by an unprecedented level of homogeneization
- Homogeneization in AI refers to the capability to build machine learning systems that can work in a wide range of applications
- In NLP, almost all of the state-of-the-art models are adapted from one of a few foundation models, such as BERT
- This phenomenon is also being observed across research communities, where transformer-based sequence modeling is being applied to images, tabular data, protein sequences, etc.

Foundation Models: Emergence

- Another distinctive feature of foundation models is emergence
- Emergence means that the behavior of a system is implicitly induced rather than explicitly constructed

Foundation Models: Emergence

- A nice emergence example would be the GPT-3 model (Brown et al. 2020) composed of 175 billion parameters
- GPT-3 permits *in-context learning*, in which the language model can be adapted to a downstream task simply by providing it with a prompt
- This ability was not specifically trained for or anticipated to arise

Why Foundation Models?

- The term describes a completely new trend in AI that is not captured by other terms
- This new trend is characterized by their sociological impact and broad shift they have introduced in AI research
- The word foundation alludes to the fact that these new models are incomplete but they constitute the basis for many task-specific models through adaptation
- The word also tries to capture the impact of the model quality on its many potential applications: is the model well or poorly constructed?

Social Impact of Foundation Models

- Foundational models have broad sociological ramifications:
    - potential exacerbation of social inequities
    - economic impact due to increased capabilities
    - environmental impact due to increased computational demands
    - concerns about misuse, legal and ethics issues

The Future of Foundation Models

- There exist significant financial motivations to expand the abilities and magnitude of foundational models
- A continuous advancement in technology in the upcoming years can be expected
- Since foundation models rely on emergent behavior, its widespread deployment should be made with caution
- Foundation models appeared almost exclusively in industry but will benefit from the guidance of academia
- The negative trend that the most powerful and latest models are no longer publicly released constitutes an important problem

References i

- Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio (Jan. 2015). "Neural machine translation by jointly learning to align and translate". In: 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.
- Bommasani, Rishi et al. (2021). "On the Opportunities and Risks of Foundation Models". In: *ArXiv*. URL: `https://crfm.stanford.edu/assets/report.pdf`.
- Brown, Tom B, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020). "Language models are few-shot learners". In: *arXiv preprint arXiv:2005.14165*.

References ii

Cho, Kyunghyun, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio (Oct. 2014). “On the Properties of Neural Machine Translation: Encoder–Decoder Approaches”. In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, pp. 103–111.

Dauphin, Yann N., Angela Fan, Michael Auli, and David Grangier (2016). “Language Modeling with Gated Convolutional Networks.”. In: *CoRR* abs/1612.08083. URL: `http://dblp.uni-trier.de/db/journals/corr/corr1612.html#DauphinFAG16`.

References iii

- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.
- Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long Short-Term Memory”. In: *Neural Computation* 9.8, pp. 1735–1780.

References iv

Kim, Yoon (Oct. 2014). “Convolutional Neural Networks for Sentence Classification”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1746–1751. DOI: 10.3115/v1/D14-1181. URL: https://aclanthology.org/D14-1181.

Mikolov, Tomas, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur (2010). “Recurrent neural network based language model.”. In: *INTERSPEECH*. Ed. by Takao Kobayashi, Keikichi Hirose, and Satoshi Nakamura. ISCA, pp. 1045–1048. URL: http://dblp.uni-trier.de/db/conf/interspeech/interspeech2010.html#MikolovKBCK10.

References v

Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever (2018). "Improving language understanding with unsupervised learning". In: *Technical Report*.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). "Sequence to Sequence Learning with Neural Networks". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems*. NIPS'14. Montreal, Canada: MIT Press, pp. 3104–3112.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need". In: *Advances in Neural Information Processing Systems*, pp. 5998–6008.

References vi

Voita, Elena, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov (July 2019). "Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 5797–5808. DOI: 10.18653/v1/P19-1580. URL: https://aclanthology.org/P19-1580.