Topic Outline / Study Guide

Pretrained Language Models

CS 5319, University of Texas at El Paso

Nigel Ward, Fall 2023

> Topics: lexical embeddings, contextual embeddings, sequence modeling, transformers, generative modeling, self-supervised learning, etc.
>
> Coverage: key concepts, historical context, when to use these representations and models, *how* to use them, how they are built. Not covered: details of model architectures or training methods, cutting-edge systems.
>
> Learning Outcomes: be able to use word embeddings and large language models with awareness of their advantages and their limitations, be able to explain roughly how they are built and trained

A. Language Models *(JM3 Ch 3 thru §3.5.1)* [80 minutes]
   a. Roles: judging likelihood of alternatives, e.g. *put it in the pin/bin*
   b. Evaluation: perplexity *(JM3 §3.2)*
   c. Historical Origin in Speech Recognition *(optionally JM**2**, §9.1)*
      i. Rule-based (knowledge-based models)
      ii. Statistical models
   d. Applications, e.g. spelling correction, selection among possible machine translation outputs, predictive text entry (auto-complete) …
   e. Classic methods: bigrams and trigrams *(JM3 §3.1)*
   f. Elaborations: domain-specific language models; context-element-retaining language models
   g. Uses as generative models *(JM3 §3.3, Assignment C2)*

B. Neural networks [15]
   a. A bunch of learned weights, like linear regression or logistic regression
   b. But more powerful, since layered, and including nonlinearities
   c. Hidden layers as intermediate representations
   d. The idea of "transfer learning" *(JM3 Ch10 Intro)*
      i. via fine tuning
      ii. via multitask learning
      iii. via additional layers ("decision heads")

C. Lexical Embeddings [40] *(JM3 Ch 11 Intro)*
   a. Feature-based (vector-space) representations of lexical semantics

i. Designed representations *(Exercise 3b)*
ii. Dense/learned/distributed representations *(JM3 Ch 6.1 – 6.4; optionally Vajjala Ch 3)*

b. Word embeddings, vector spaces, the distributional hypothesis
c. Idea: a feature-set representation that is useful for all sorts of things
d. Idea: a magic number sequence that has certain properties (minimizes a certain loss)
e. Word2vec and training via skip-grams *(JM3 §6.8, BG)*
   i. Handling unknown words via word pieces, byte-pair encoding, eg fasttext
f. This is "self-supervised learning", not supervised, nor unsupervised
g. Ways to use pre-trained word embeddings, e.g. via gensim
h. Applications: sentiment analysis, other classification tasks, IR, question answering …
i. Assignment F
   i. Glove vs Word2vec; euclidean distances vs cosine similarity
   ii. Should we be surprised if some plurals are not close to the singular forms? E.g. *stocks/stock, sweets/sweet*

D. Sequence-to-sequence modeling *(optionally JM3 Ch 9, and 8)* [20]

a. The need
b. Speech recognition; to recognize the middle of a /b/ need the first part of the /b/
   i. CNNs (fixed-width left context) … but fixed lengths may not work
   ii. c.f. HMMs … designing just the right number of states
c. Recurrent neural networks (RNNs) and the idea of hidden state
   i. Illustrate with POS tagging,
      1. as in <s> *so long, and thanks for all the fish*
      2. as for *model* in *large langauge model*
   ii. digression: c.f. dependency parsing, as in *large language model* … what does the word *large* modify … may affect gender in some languages

d. Gating
   i. long short-term memory networks (LSTMs) and with
   ii. Gated Recurrent Units (GRUs)
   iii. Occasionally bidirectional (BiLSTMS)
e. Encoder-decoder networks, often with a single hidden state
f. From deciding to keep/forget context, to deciding what context matters, even distant *(PICS)*
g. Attention, transformers

E. Contextual lexical embeddings [25]

a. The need: word meanings are context-dependent
b. Innovation: a "masked language model pretraining objective"

        i. Similar to what we saw in word2vec

c. BERT: Bidirectional Encoder Representations from Transformers

d. What BERT gives you: word representations that are great for prediction … of many things

    i. Input: a sequence of words or word pieces

        1. The latter in order to handle, e.g. *fabulicious, cactuses*

    ii. Output: a sequence of vectors

    iii. People usually average all the word vectors, or use the CLS vector

e. How BERT and similar systems are built:

    i. Gating, attention, and transformers, again *(optionally JM3 §11.1)*

    ii. Training/masking *(optionally JM3 §11.2)*

F. Generative modeling *(JM3 §10.2 last paragraphs)* [30]

a. Large Language Models (LLMs) (*watch ILLM)*

b. Generative Pretrained Transformer (GPT) models: GPT, PaLM, LLaMA …

c. These are large and expensive (to build, to run), although small and distilled versions exist

    i. Up to 1 trillion parameters (Is more better?)

    ii. Up to 1 trillion tokens (Is more better?)

d. Evaluation

    i. Perplexity/cross-entropy and other loss functions

    ii. Performance on composite benchmarks, e.g. Super-GLUE (logic puzzles, reading comprehension, anaphor, disambiguation, etc.)

        1. Often outperforming humans

e. Perspectives

    i. These are just lossy compressions of the internet

        1. With hallucinations

    ii. These can exhibit emergent abilities, such as simple arithmetic

    iii. They are "foundation models"

f. Three variants

    i. Raw (genertic predictive)

    ii. Prompt-tuned

    iii. Dialog-tuned, based on reinforcement learning with human judges

g. Your options

    i. Access the chatbot versions {ChaptGPT, Bard, etc.}

        1. optimized (trained) for long-form dialog (sometimes by human trainers/judges)

        2. have filters to "hide" biases

    ii. Use the APIs and add a "decision-head" layer

    iii. Fine tune (with even relatively modest data

iv. Train your own
v. Write prompts *(FK)*
1. Style tips help a lot
2. Various levels of instruction/specificity: zero-shot, one-shot, few-shot, e.g. giving question-answer pairs to illustrate
3. Prompt engineering for automatic prompting and higher accuracy
4. Preferably use instruction-tuned LLMs

h. Applications: text classification, question answering, text generation, summarization, paraphrasing

G. Pretrained models beyond text alone
a. pretrained models for speech
b. cross-modal models, with images, video, code
c. multilingual models

References:

JM3: *Speech and Language Processing, 3rd edition*, by Jurafsky and Martin https://web.stanford.edu/~jurafsky/slp3/

JM2: ditto second edition (JM2)

BG: Word2Vec Algorithm, Jordan Boyd-Graber, first 8 minutes https://www.youtube.com/watch?v=c3yRH0XZN2g

Vajjala: Vajjala, Sowmya, et al. *Practical natural language processing: A comprehensive guide to building real-world NLP systems*. O'Reilly Media, 2020.

ILLM: Introduction to Large Language Models, by Google Cloud Tech, on Youtube, first 14 minutes https://www.youtube.com/watch?v=zizonToFXDs

FK: Prompt Engineering Complete Guide, Fareed Khan. https://medium.com/@fareedkhandev/prompt-engineering-complete-guide-2968776f0431

PICS: second image at https://www.tensorflow.org/text/tutorials/nmt_with_attention ; second image at https://en.wikipedia.org/wiki/Large_language_model ; second image set at https://devopedia.org/attention-mechanism-in-neural-networks