Language Models

A *Language Model* is an instance of the "predict the next" paradigm where

- given a sequence of words
- we try to predict the next word

Recall the architecture to solve "predict the next word" and data preparation

Language Modeling task

Architecture	Data preparation

The raw data

- e.g., the sequence of words $\mathbf{s} = \mathbf{s}_{(1)}, \ldots \mathbf{s}_{(\bar{T})}$

is not naturally labeled.

We need a Data Preparation step to create labeled example i

$$\begin{aligned} \mathbf{x^{(i)}} &= \mathbf{s}_{(1)}, \ldots \mathbf{s}_{(i)} \\ \mathbf{y^{(i)}} &= \mathbf{s}_{(i+1)} \end{aligned}$$

We have called this method of turning unlabeled data into labeled examples: *Semi-Supervised* Learning.

In the NLP literature, it is called *Unsupervised Learning*.

There are abundant sources of raw text data

- news, books, blogs, Wikipedia
- not all of the same quality

The large number of examples that can be generated facilitates the training of models with very large number of weights.

This is extremely expensive but, fortunately, the results can be re-used.

- Someone with abundant resources trains a Language Model on a broad domain
- Publishes the architecture and weights
- Others re-use

Predict the next ? Really: predict the *distribution* of next

We have casually defined the Language Modeling objective as predicting the next token.

As you can see: the head layer is a Classifier

- produces a probability for *each token* in the vocabulary as being the next
- We choose one token by sampling from this probability distribution
    - Greedy sampling: always chose the token with highest probability
    - Non-greedy sampling

The Masked Language Modeling objective

There is a variation on the Language Modeling objective called the *Masked Language Modeling* objective.

- Language Modeling objective: given $s[1:t-1]$, predict $s[t]$
- Masked Language Modeling objective
    - Given $s[1:t]$
    - Randomly chose an index $1 \leq j \leq t$
    - "Mask" token j by replacing it with <MASK> so that the input becomes
    $$s_{(0)}, \ldots s_{(j-1)}, <\text{MASK}>, s_{(j+1)}, \ldots s_{(t)}$$
    - Predict the value behing the mask, e.g., $s_{(j)}$
- The Language Modeling objective is the special case where $j = t$

Unsupervised Pre-Training + Supervised Fine-Tuning (Transfer Learning)

How do we adapt a Language Model to solve other Target tasks ?

The obvious answer is via Transfer Learning

- The Language Model has learned a lot about the nature of language
    - perhaps the language-knowledge can be transfered to a new task
- Replace the "head" that predicts the next token
- With a new task-specific head
- Train the new model on labeled examples from the Target task
    - the task-specific head **must** be trained
    - the language-model weights **can** (but don't have to) be adapted

This paradigm is called *Unsupervised Pre-Traininng + Supervised Fine-Tuning*.

Example: Fine-Tuning a Pre-trained Language Model to sentiment

This is a straight-forward application of Transfer Learning

- Replace the Classification Head used for Language Modeling
    - e.g., a head that generated a probability distribution over wo vocabulary
- By an untrained Binary Classification head (Positive/Negative sentim
- Train on examples. Pairs of
    - sentence
    - label: Positive/Negative

Other uses of a Language Model: Feature ba
Transfer Learning

We can generalize the procedure of "replacing the head": re-use the featu
by the Source model.

Let $f_\Theta(\mathbf{x})$ denote the function computed by the Source Language Model c
sequence $\mathbf{x}$

- the output of the layer before the final Classification layer that tran
  into the token Vocabulary
- the Source model is parameterized by Θ

Feature based Transfer Learning computes

$$g_\Phi(f_\Theta(\mathbf{x}))$$

for the function g (parameterized by Φ) computed by a new NN.

The case where g_Φ is a Classifier is the special case of creating a new Targe
head.

For example, the representations produced by an Encoder Transformer tra
Language Modeling task

- embody great semantic meaning: meaning of a word in the context
  predecessor words
- are much shorter than One Hot Encoding of tokens
- are more meaningful than simple Word Embeddings
    - embody great semantical meaning

We may transfer the "features" that these representations encode for the
creating *Context Sensitive Representations* of words.

For example, the meaning of the word "bank" varies with context

- a financial institution
- the land adjacent to a river
- tilting the wings of an airplane

Word Embeddings are a unique representation of a word, independent of context. So they cannot fully capture the multiple meanings of the word "bank"

Using a bi-directional Encoder, the representation becomes a *Context Sensitive Representation*

- able to distinguish the multiple meanings of the word "bank"
- from the context (preceding and succeeding words) in which it appears

See the [ELMo paper (https://arxiv.org/abs/1802.05365)](https://arxiv.org/abs/1802.05365)

Multi-task learning

One area of deep interest is *multi-task learning*

- Training a model to implement multiple tasks

A model that implements a single task computes

$$p(\text{output} \mid \text{input})$$

A model that implements several tasks computes

$$p(\text{output} \mid \text{input}, \text{task-id})$$

An interesting special case

- the representation of the first or final token in the sequence
    - usually a special token <START>, <END>
- is a summary of the *entire sequence* (using a bi-directional RNN or no
  Encoder Transforer

Using the Universal API, a Language Model may be adapted to solve *multipl*
simultaneously.

This requires you to construct a training set
This representation is a very short encoding of a long sequence

- with examples from each task
- like hashing
- where each example has an additional "feature" that identifies the ta
  applies

One can use this to implement *Semantic Search*

For example

- "hash" each page on the Web
  "(Translate to French, English text, French