Tutorial

Language Modelling

Teaching Assistant,
Somesh Kumar Singh,
2018A7PS0175G

Aim

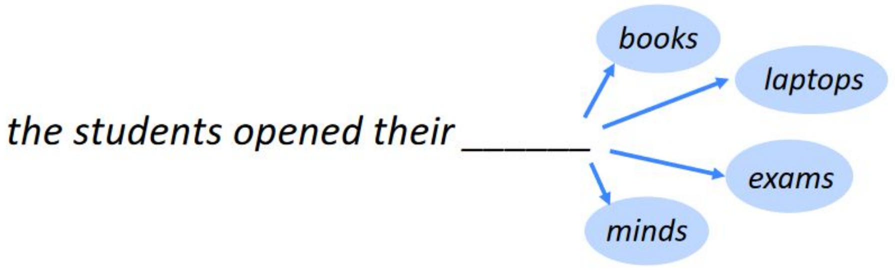
- Understand the concepts of Language modelling
- Practical aspects
- Hands on practice
- Introduction to modern Language modeling
- Introduction to SOTA (State of the art)

Plan of work

1. Introduction
    1.1. What is Language Modelling
    1.2. Uses of Language Models
    1.3. Levels of Language Modelling
    1.4. Types of Language Modelling
2. Statistical Language Modelling
    2.1. Markov-Assumption
    2.2. Smoothing
    2.3. Evaluation
    2.4. Limitations
3. Neural Language Modelling
    3.1. Word Vectors
    3.2. Context Window
    3.3. Generalization
4. Language Modelling for Text Generation (If time permits!)
5. Diving Deeper

1.1 Introduction: What is Language Modelling

Language Modeling is the task of predicting what word comes next.



$$P(\boldsymbol{x}^{(t+1)} \mid \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})$$

More formally: given a sequence of words compute the probability distribution of the next word from a given vocabulary.

1.1 Introduction: What is Language Modelling

Language Modeling Language modeling is the task of assigning a probability to sentences in a language (“what is the probability of seeing the sentence the lazy dog barked loudly?”).

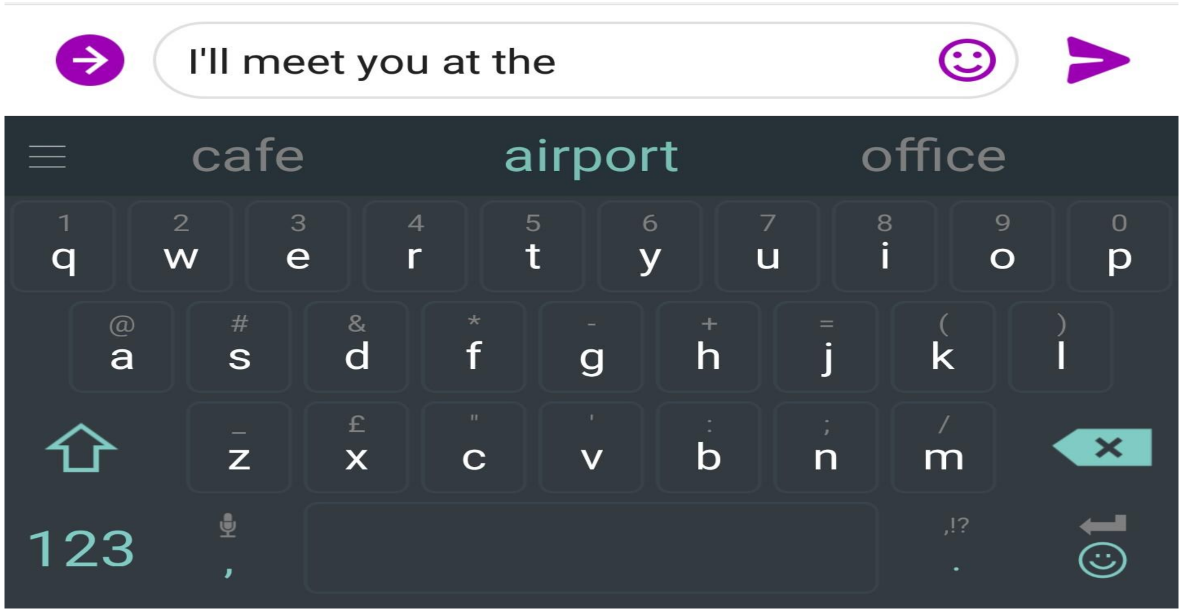
A big brown bear! -> Likely

Bear brown A! -> Unlikely

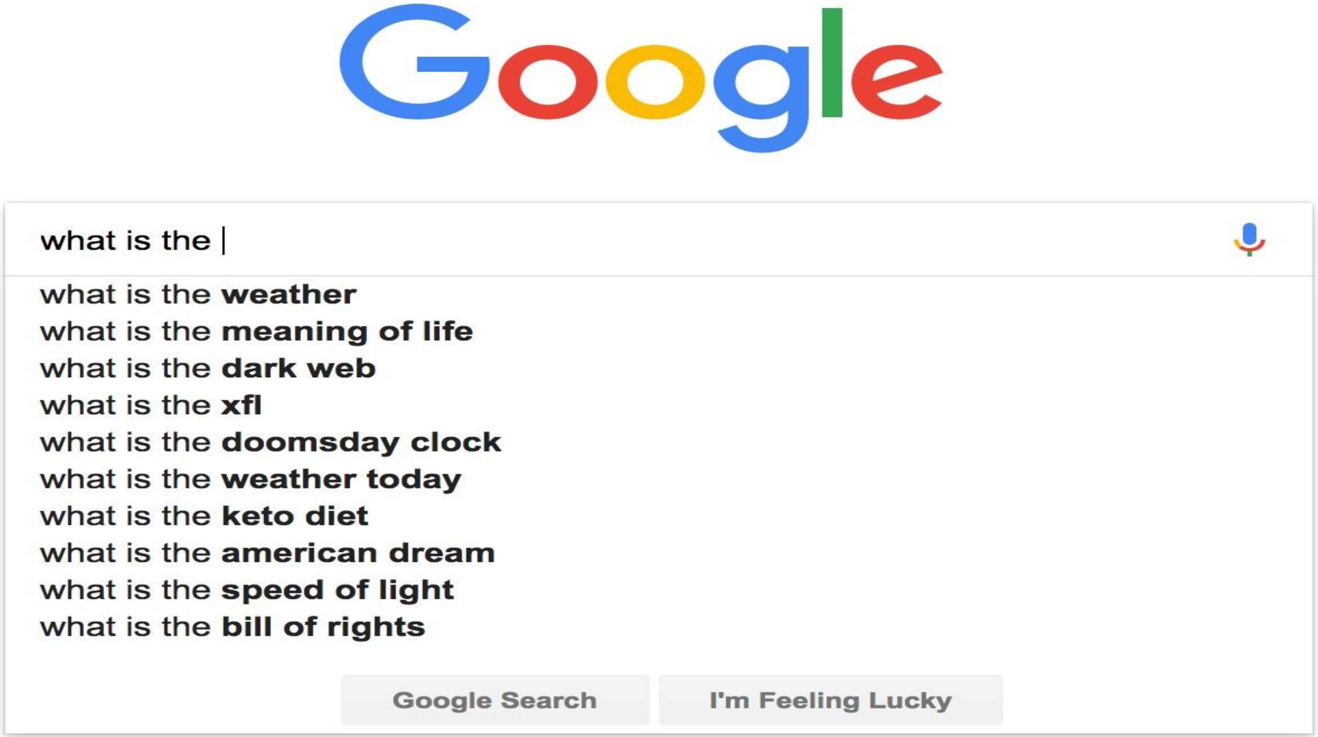
More formally: assign a probability to any sequence of words ($w_{1:n}$), i.e., to estimate $P(w_{1:n})$ Using the chain-rule of probability, this can be rewritten as:

$$P(w_{1:n}) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_{1:2})P(w_4 \mid w_{1:3}) \dots P(w_n \mid w_{1:n-1}).$$

1.2 You use Language models everyday!


I'll meet you at the

1.2 You use Language models everyday!



what is the

what is the **weather**
what is the **meaning of life**
what is the **dark web**
what is the **xfl**
what is the **doomsday clock**
what is the **weather today**
what is the **keto diet**
what is the **american dream**
what is the **speed of light**
what is the **bill of rights**

1.3 Levels of Language Modelling

1. Character Language Modelling
2. Word Language Modelling

1.4 Types of Language Modelling

1. Statistical Language Modelling
2. Neural Language Modelling

2. Statistical Language Model: N-gram Language Model

- **Question**: How to learn a Language Model?
- **Answer** (pre- Deep Learning): learn a *n*-gram Language Model!

- Definition: A *n*-gram is a chunk of *n* consecutive words.
  - unigrams: “the”, “students”, “opened”, ”their”
  - bigrams: “the students”, “students opened”, “opened their”
  - trigrams: “the students opened”, “students opened their”
  - 4-grams: “the students opened their”

- Idea: Collect statistics about how frequent different n-grams are, and use these to predict next word.

2.1 Statistical Language Model: Markov Assumption

While the task of modeling a single word based on its left context seem more manageable than assigning a probability score to an entire sentence, the last term in the equation still requires conditioning on (n-1) words, which is as hard as modeling an entire sentence.

For this reason, language models make use of the markov-assumption, stating that the future is independent of the past given the present. More formally, a n^{th} order markov-assumption assumes that the next word in a sequence depends only on the last n words.

2.1 Statistical Language Model: Markov Assumption

First we make a simplifying assumption the next word depends only on the preceding n-1 words.

$$P(\boldsymbol{x}^{(t+1)}|\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(1)}) = P(\boldsymbol{x}^{(t+1)}|\overbrace{\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)}}^{n\text{-1 words}})$$

$$\begin{array}{l} \text{prob of a n-gram} \longrightarrow \\ \text{prob of a (n-1)-gram} \longrightarrow \end{array} = \frac{\boxed{P(\boldsymbol{x}^{(t+1)},\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)})}}{\boxed{P(\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)})}}$$

2.1 Statistical Language Model

Question: How do we get these n-gram and (n-1)-gram probabilities?
Answer: By counting them in some large corpus of text!

$$P(w_n|w_1^{n-1}) = \frac{C(w_1^{n-1}w_n)}{\sum_w C(w_1^{n-1}w)}$$

While the kth order markov assumption is clearly wrong for any k (sentences can have arbitrarily long dependencies, as a simple example consider the strong dependence between the first word of the sentence being what and the last one being ?), it still produces strong language modeling results for relatively small values of k, and was the dominant approach for language modeling for decades.

2.2 Statistical Language Model: Smoothing

- **Zero Probability events**

One way of avoiding zero-probability events is using smoothing techniques, ensuring an allocation of a (possibly small) probability mass to every possible event. e simplest example is probably additive smoothing, also called add smoothing. Let's take a look!

2.3 Statistical Language Model: Evaluation

- **Perplexity**

An intrinsic evaluation of language models is using perplexity over unseen sentences. Perplexity is an information theoretic measurement of how well a probability model predicts a sample. Low perplexity values indicate a better fit.

Good language models (i.e., reflective of real language usage) will assign high probabilities to the events in the corpus, resulting in lower perplexity values. Perplexity measure is a good indicator of the quality of a language model.

$$2^{-\frac{1}{n}\sum_{i=1}^{n}\log_2 \mathrm{LM}(w_i|w_{1:i-1})}.$$

2.4 Statistical Language Model: Limitations

1. Back off / Long History Problems.
2. Larger N-grams
3. Lack of Generalisation

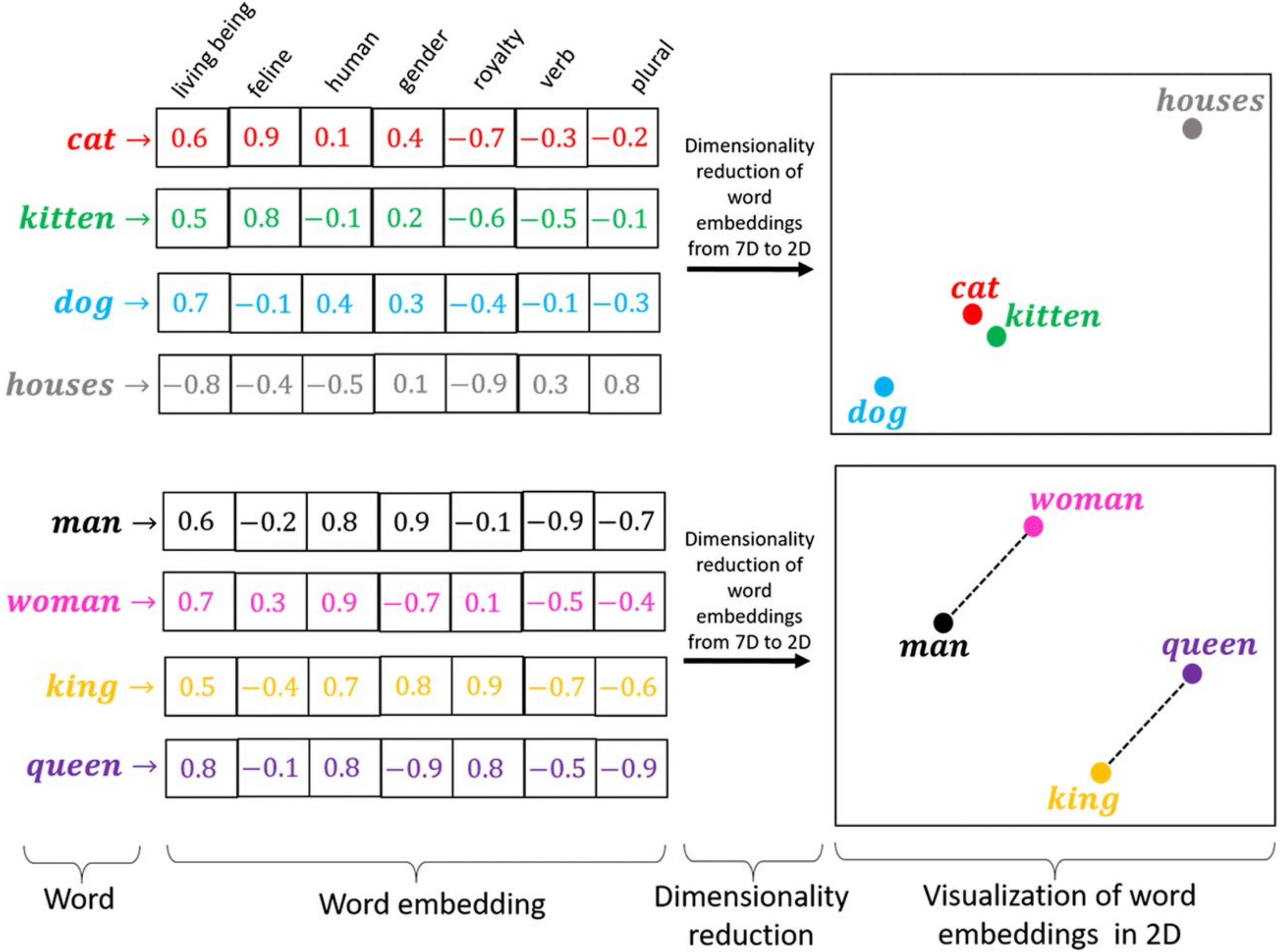
3.1 Neural Language Modelling: Word Vectors

In modern times, NLMs are mostly used for Language Modelling especially where longer dependencies are to be captured. Few core concepts required to understand them are.

1. Word Embeddings
2. MLP / Neural Networks

Since these topics are beyond the scope of the discussion, we will make simple assumptions.

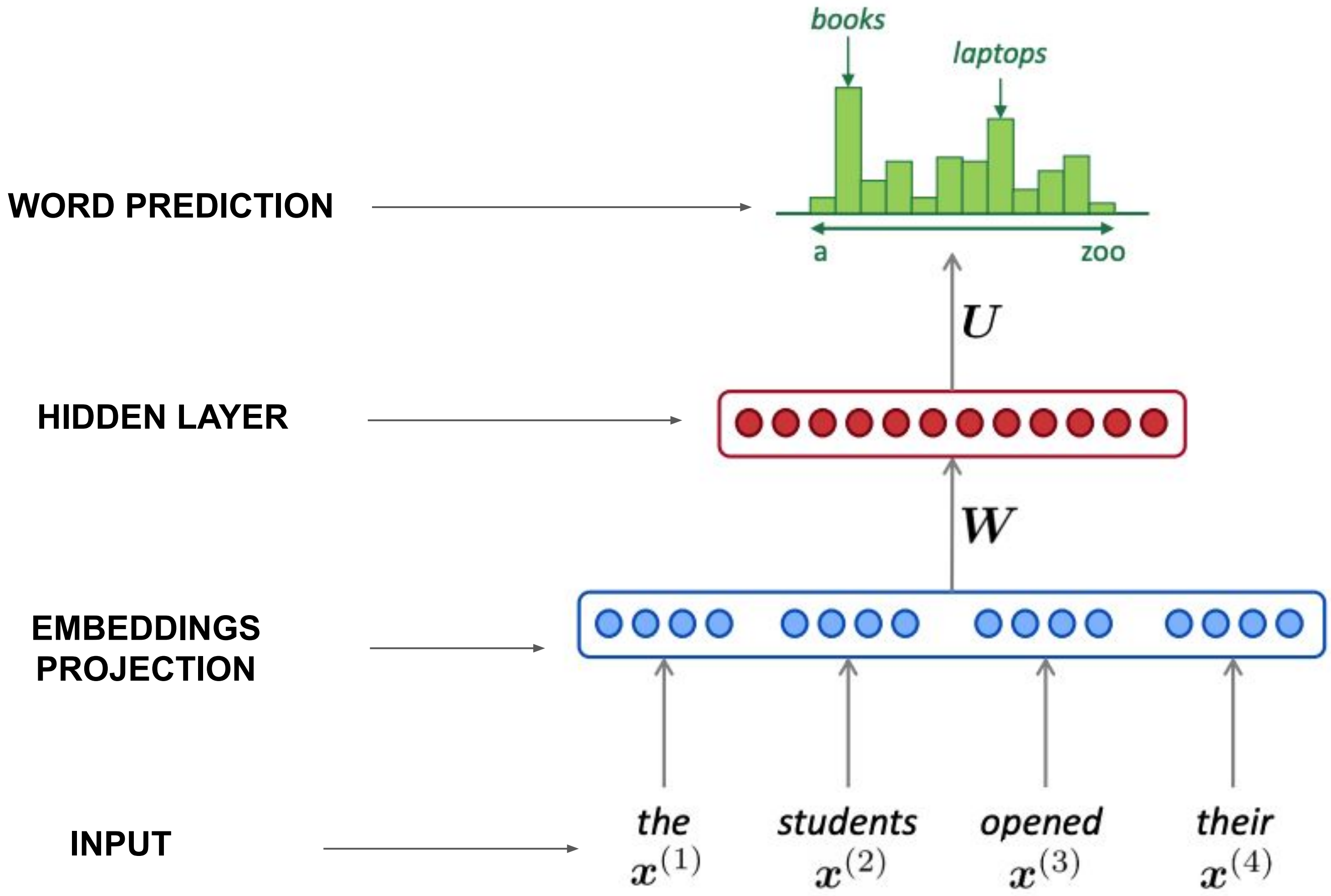
In SLMs, words were treated (or mapped) as fixed integers, A:0, An: 1, The:2 and so on, in modern NLMs words are replaced by vectors (embeddings) this gives a continuous representation for optimization.



Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D

	living being	feline	human	gender	royalty	verb	plural
cat →	0.6	0.9	0.1	0.4	−0.7	−0.3	−0.2
kitten →	0.5	0.8	−0.1	0.2	−0.6	−0.5	−0.1
dog →	0.7	−0.1	0.4	0.3	−0.4	−0.1	−0.3
houses →	−0.8	−0.4	−0.5	0.1	−0.9	0.3	0.8

man →	0.6	−0.2	0.8	0.9	−0.1	−0.9	−0.7
woman →	0.7	0.3	0.9	−0.7	0.1	−0.5	−0.4
king →	0.5	−0.4	0.7	0.8	0.9	−0.7	−0.6
queen →	0.8	−0.1	0.8	−0.9	0.8	−0.5	−0.9


WORD PREDICTION
HIDDEN LAYER
EMBEDDINGS PROJECTION
INPUT
books
laptops
a
zoo
U
W
the
students
opened
their
$x^{(1)}$
$x^{(2)}$
$x^{(3)}$
$x^{(4)}$

3.2 Neural Language Modelling: Context Window

Nonlinear neural network models solve some of the shortcomings of traditional language models: they allow conditioning on increasingly large context sizes with only a linear increase in the number of parameters.

- N-gram Model complexity with increasing N -> $|V|^N$
- NLM Complexity with increasing window size N -> Linear in N (Concatenation)

This solves both, Larger N gram and Longer Dependency problems.

3.3 Neural Language Modelling: Generalization

NLMs optimize the embedding space to get the MLE for the target word, therefore it learns disentangled representations instead of class co-occurrence.

Consider the example:
Red car, Blue car

4. Language Generation

After training a language model on a given collection of text, one can generate (“sample”) random sentences by beginning from the Start Token, take a random window sized sample or padding and predict the the second word conditioned on the first, and so on, until predicting the end-of-sequence symbol.

1. Greedy Generation
2. Beam Search

5. Further?

NLMs have progressed enormously from here on, MLP were replaced by RNNs, LSTMs, attention and transformers for the modern age NLP.

You have gained sufficient knowledge to go through this if you want to: https://towardsdatascience.com/bert-for-dummies-step-by-step-tutorial-fb90890ffe03