Introduction to Neural Language Models

Abolfazl Mahdizade

Outline

- *Language Model*
- *Neural Language Model*

What is Language Model (LM)

- Language Model (LM) assign probability values to sequences of words
- Language Model is a fundamental part of many systems
  - Machine translation
  - Spelling corrections
  - Automatic sentence completion
  - Summarization
  - Question Answering
  - Speech recognition
  - ...

Language Model

- *Probability of observing an entire sentence:*

$$p(w_1, w_2, \ldots w_t) = p(w_1)p(w_2|w_1)\ldots p(w_t|w_{t-1}, \ldots w_1)$$

- *Estimating these probabilities can be tough*
- *Language models seek to predict the probability of observing next word given the previous words*

$$p(w_{t+1}|w_1, w_2, \ldots w_t)$$

Language Model (Continue)

- *Maximum likelihood estimate*

$$p(x_{t+1}|x_1,\ldots x_t) = \frac{count(x_1,x_2,\ldots x_t,x_{t+1})}{count(x_1,x_2,\ldots x_t)}$$

- *Not enough data* $\Rightarrow$ *Markov assumption*
- *The Markov assumption*
  - *the probability of observing a word at a given time is only dependent on the word observed in the previous time step*

$$p(x_{t+1}|x_1,x_2,\ldots x_t) = p(x_{t+1}|x_t)$$

Language Model (Continue)

- *The probability of a sentence with Markov assumption*

$$p(w_1, w_2, \ldots w_t) = p(w_1) \prod_{i=2}^{t} p(w_i | w_{i-1})$$

- *The Markov assumption can be extended to condition the probability of the previous two, three, four, and so on words*
- *This is where the name of the n-gram model comes in*
  - *n is the number of previous timesteps*

Language Model (Continue)

- *The unigram model*

$$p(x_{t+1}|x_1, x_2, \ldots x_t) = p(x_{t+1})$$

- *The bigram model*

$$p(x_{t+1}|x_1, x_2, \ldots x_t) = p(x_{t+1}|x_t)$$

- …

What is Neural Language Model (NLM)

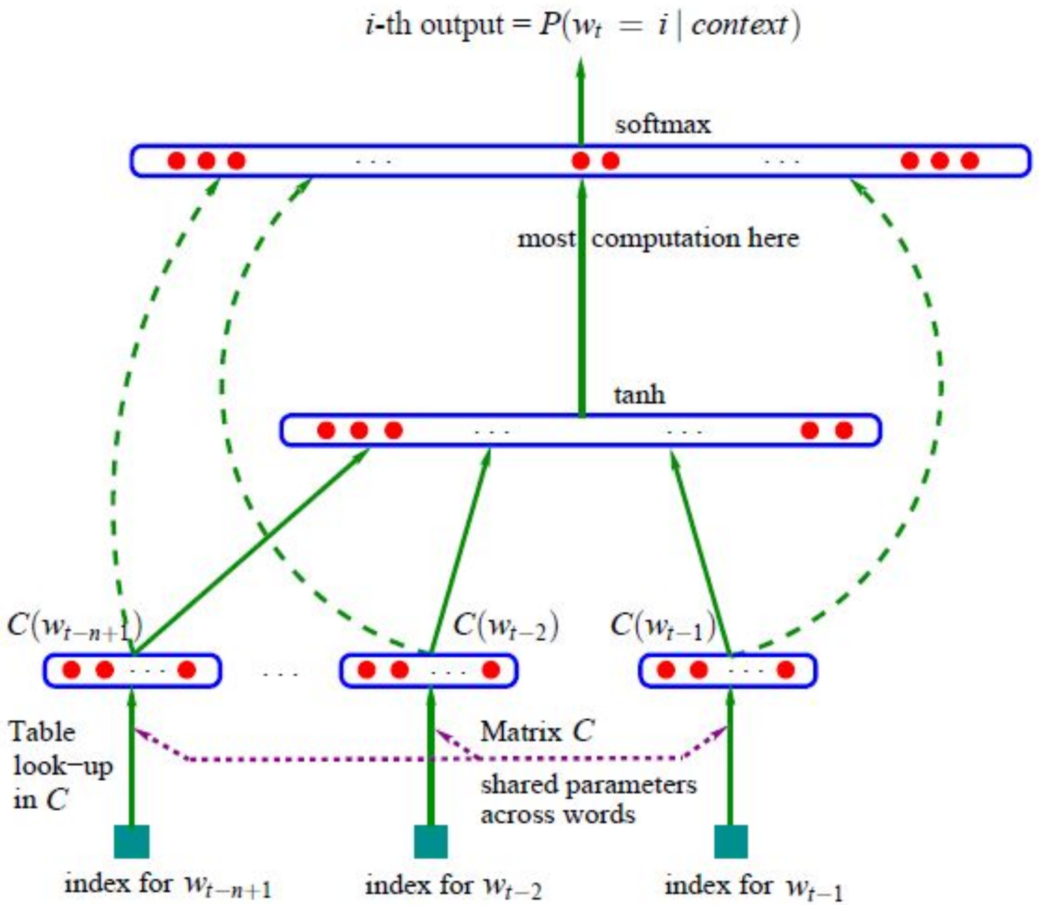
- A neural network language model is a language model based on Neural Networks
- Currently, all state of the art language models are neural networks
- Type of NLMs
  - Feed-Forward (like Convolution)
  - RNNLM (LSTM Networks)

Neural Language Model

Neural Language Model

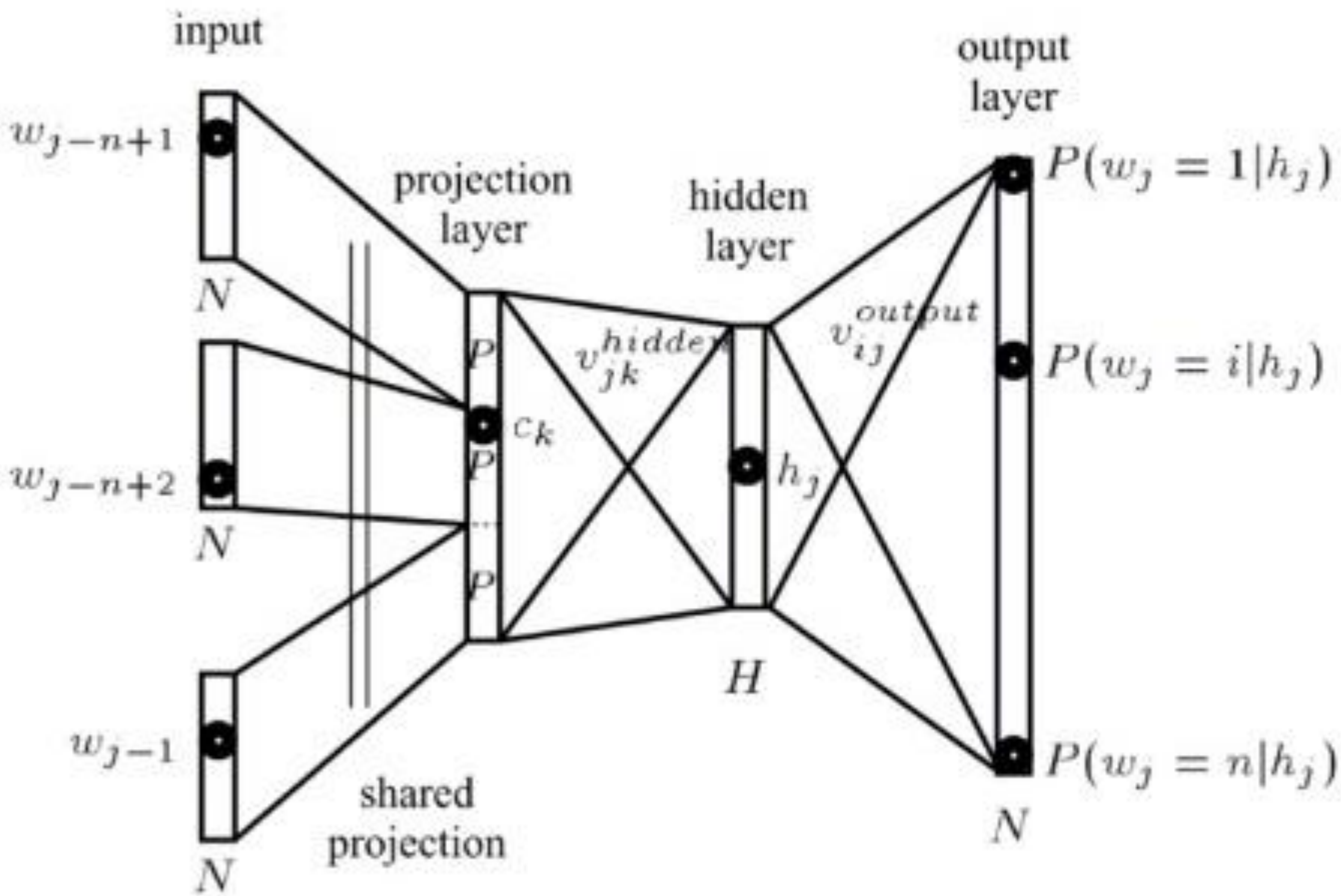
- *Takes words from a vocabulary as input (One-hot vector)*
  - *Sparse representations of words in a vocab-size vector space*
- *Embeds words as vectors into a lower dimensional space (Word Embeddings)*
  - *Dense representations of words in a low-dimensional vector space*
- *Word Embeddings = Word Vectors = Distributed Representations*
- *Neural Word Embeddings*
  - *word embeddings learned by a neural network (backpropagation)*

Neural Language Model (Continue)



Classic neural language model (Bengio et al., 2003)

Neural Language Model (Continue)



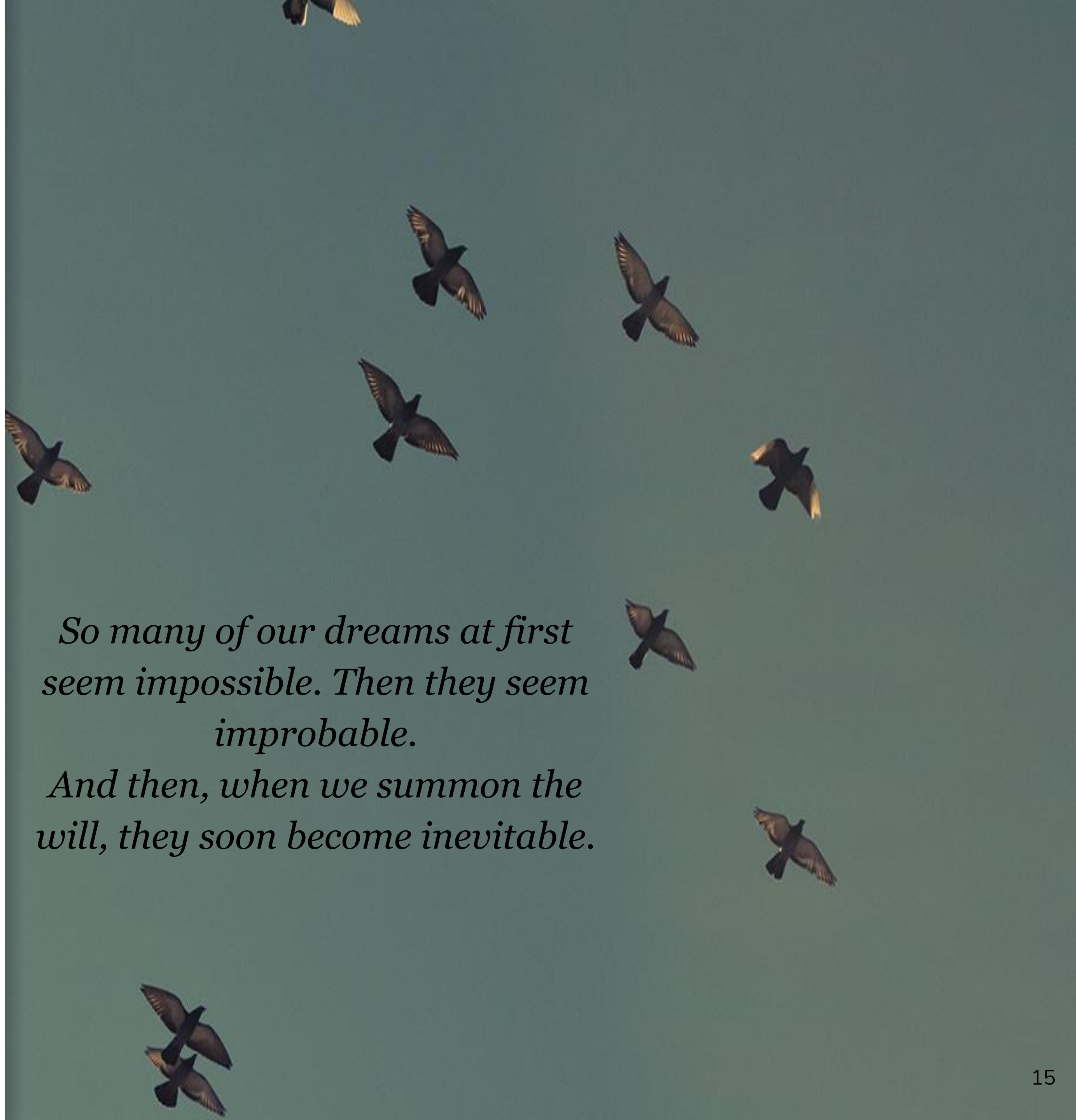
A neural language model (Bengio et al., 2006)

Neural Language Model (Continue)

- *Training a Neural Language Model*
  - *Corpus*
  - *Vocab (from corpus) and vocab size |V|*
  - *Cutoff words (use as unknown <UNK>)*
  - *Padding (SOS <S>, EOS </S>, …)*
  - *Embeddings*
    - *Static (Word2Vec)*
    - *Dynamic (Embedding Layer)*

Neural Language Model (Continue)

- *Embedding Layer*
  - *Layer that generates word embeddings by multiplying an index vector with a word embedding matrix*
- *Intermediate Layer(s)*
  - *One or more layers that produce an intermediate representation of the input, (fully-connected, Convolution, LSTM, …) that applies a nonlinearity to the concatenation of word embeddings of n previous words*
- *Softmax Layer*
  - *the final layer that produces a probability distribution over words in V*


So many of our dreams at first seem impossible. Then they seem improbable.
And then, when we summon the will, they soon become inevitable.