>>> Introduction to Natural Language Processing

Name: Khairi Abidi[†]
Date: April 6, 2022

[†] abidikhairi.1337@gmail.com

\>>> Table of Contents

1. Introduction

2. Words Embedding

3. Word2Vec

4. Recurrent Neural Networks

>>> Machine Learning Tools Used In this Tutorial[1]

* torch v1.9.0
* torchtext v0.10.0
* nltk v3.6
* gensim v4.1.2
* matplotlib v3.4
* numpy v1.22
* CUDA driver v11.4

[1] code available at: https://github.com/abidikhairi/introduction-nlp

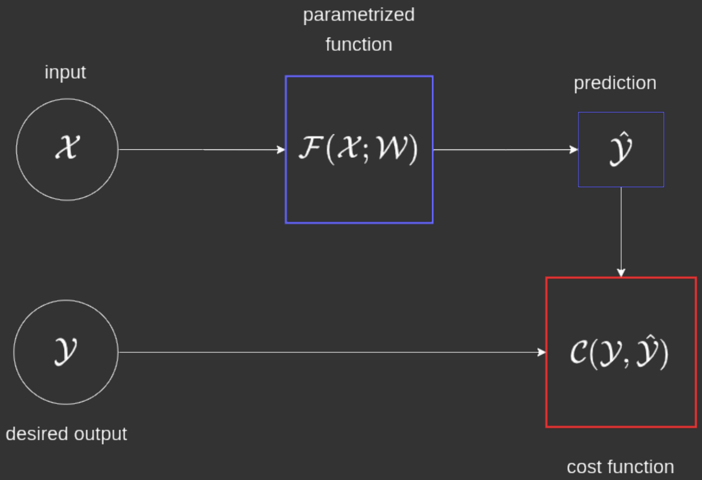


Figure: training neural networks

>>> Introduction

Natural language processing is the intersection of linguistics, computer science, and artificial intelligence.

>>> Introduction

Its principal purpose is to design computer programs that can understand natural language. Such as understanding the semantic of phrases.

>>> Introduction

for the sake of understanding natural language. we next introduce the words embedding techniques.

>>> Embbeding: Definition

Before going deeper into word embedding methods, before all else, we outline the notion of embedding.

>>> Embedding: Definition

Mathematically speaking, an embedding is a mapping $\mathcal{G}$ from one space E to another space F. in such a way that $dim(F) << dim(E)$. and $\mathcal{G}$ must preserve the structure of E.

$$\mathcal{G} : E \longmapsto F \tag{1}$$

>>> Embedding: Machine Learning

From the perspective of machine learning, embedding is considered a dimensionality reduction technique. In the following, we present some experiments on dimensionality reduction using TSNE [10].

>>> Embedding: MNIST dataset

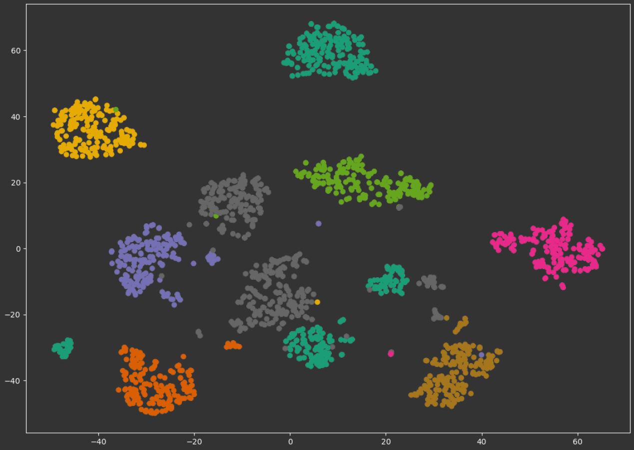


Figure: Embedding of MNIST dataset [4]

\>>> Embedding: Iris dataset

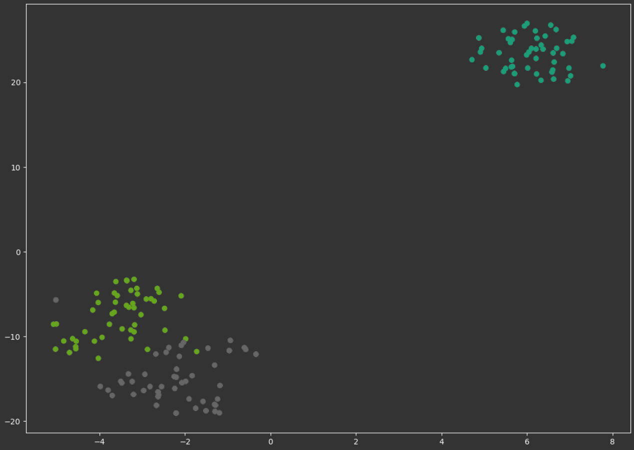
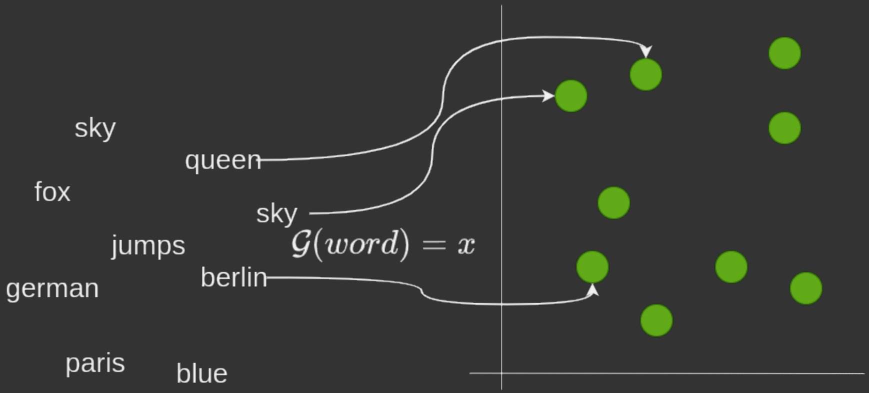


Figure: Embedding of Iris dataset [9]

>>> Word Embedding

Word embedding is a technique applied by text processing models to transform the text into real-valued vectors by capturing the semantic and syntactic relations between words. Later, those vectors will be in use by downstream tasks. Such as sentiment analysis [7], [8] or fake news detection [3].

>>> Word Embedding



$$\mathcal{G} : E \longmapsto \mathbb{R}^2$$

Figure: Example of word embedding

>>> Word Embedding: Complete Process

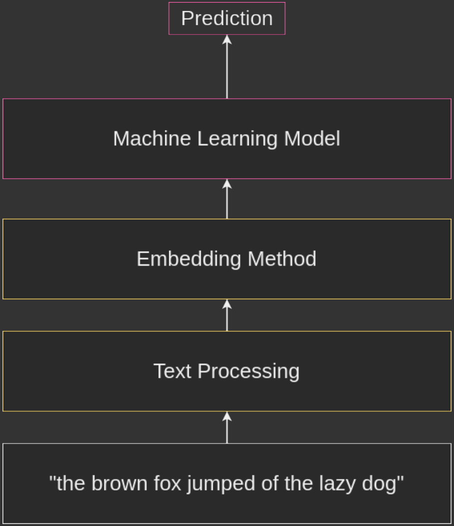


Figure: Machine learning on Text

>>> Word Embedding: Methods

* Latent Semantic Analysis [2] (Matrix Factorization)
* Term-Frequency Inverse Document Frequency [1] (Frequentist Approach)
* Word2Vec [5] (Probabilistic Neural Network Model)

>>> Word2Vec

In 2013, Tomas Mikolov (Google Researcher) proposed a simple and efficient neural network architecture called Word2vec [5] for estimating word representations (embedding).

Word2Vec estimates word representation by maximizing the likelihood of seeing a context given the center word.

$$\underset{\theta}{\operatorname{argmax}}\, log(p(w_c|w_t;\theta)) \tag{2}$$

>>> Word2Vec: SkipGram

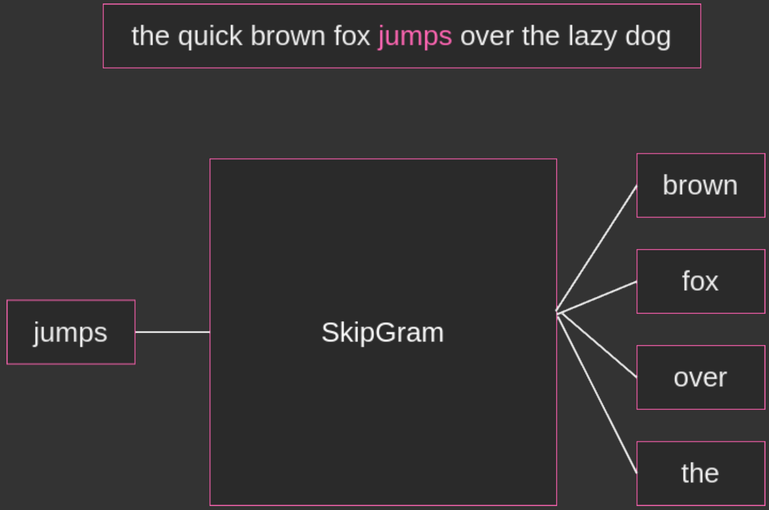


Figure: Skip-Gram Architecture

>>> Word2Vec: Probability Estimation Function

Word2Vec defines the probability function to be the softmax function.

$$p(w_c|w_t) = \frac{exp(w_c.w_t)}{\sum_{i=0}^{W} exp(w_i.w_c)} \tag{3}$$

>>> Word2Vec: Softmax

As we note, the time complexity of the softmax grows exponentially as the dataset grows. Given that text, datasets have 10^6 - 10^9 words. Therefore, the computation of the softmax functions becomes very expensive.

>>> Word2Vec: Noise Contrastive Estimation

In [6], Tomas Mikolov presented new methods to overcome the exorbitance cost of the softmax function.

$$log\ \sigma(w_c^T.w_t) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P(n)} \left[log\ \sigma(-w_i^T.w_t)\right] \tag{4}$$

>>> Recurrent Neural Networks

Recurrent Neural Networks are class of Deep Learning architectures that operates on sequential inputs, such as time series, text, and voice.

>>> RNN Cell

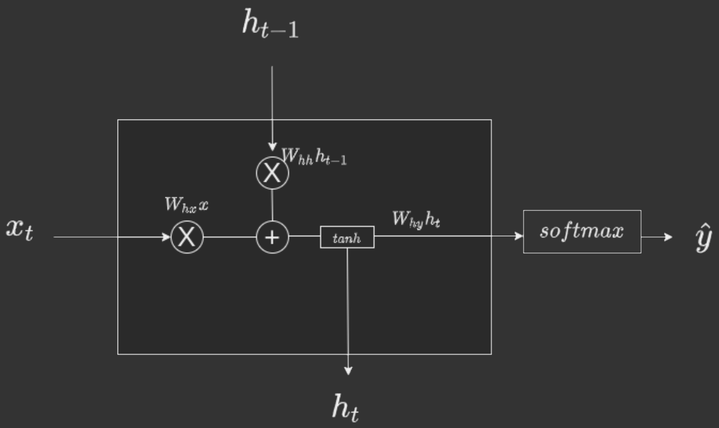


Figure: RNN Cell

>>> RNN Layer

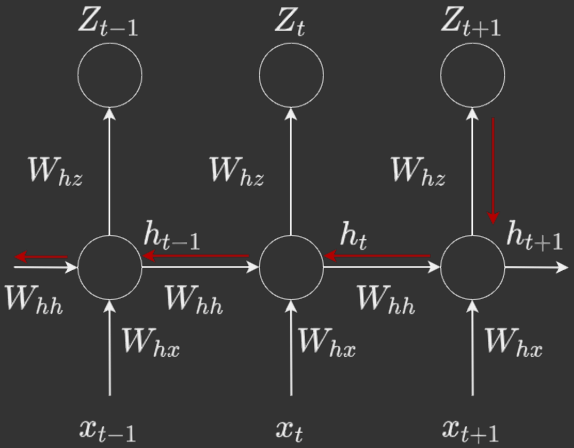


Figure: RNN Layer

>>> References I

TF-IDF.
In C. Sammut and G. I. Webb, editors, *Encyclopedia of Machine Learning*, pages 986--987. Springer US.

P. W. Foltz.
Latent semantic analysis for text-based research.
28(2):197--202.

N. Islam, A. Shaikh, A. Qaiser, Y. Asiri, S. Almakdi, A. Sulaiman, V. Moazzam, and S. A. Babar.
Ternion: An autonomous model for fake news detection.
11(19):9292.
Number: 19 Publisher: Multidisciplinary Digital Publishing Institute.

>>> References II

- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner.
  Gradient-based learning applied to document recognition.
  In *Proceedings of the IEEE*, volume 86, pages 2278--2324, 1998.
- T. Mikolov, K. Chen, G. Corrado, and J. Dean.
  Efficient estimation of word representations in vector space.
- T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean.
  Distributed representations of words and phrases and their compositionality.

>>> References III

- S. Pal, S. Ghosh, and A. Nag.
  Sentiment analysis in the light of LSTM recurrent neural networks.
  9:33--39.
- A. Patel and A. K. Tiwari.
  Sentiment analysis by using recurrent neural network.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay.
  Scikit-learn: Machine learning in Python.
  Journal of Machine Learning Research, 12:2825--2830, 2011.

>>> References IV

L. van der Maaten and G. Hinton.
Visualizing data using t-SNE.
Journal of Machine Learning Research, 9:2579--2605, 2008.