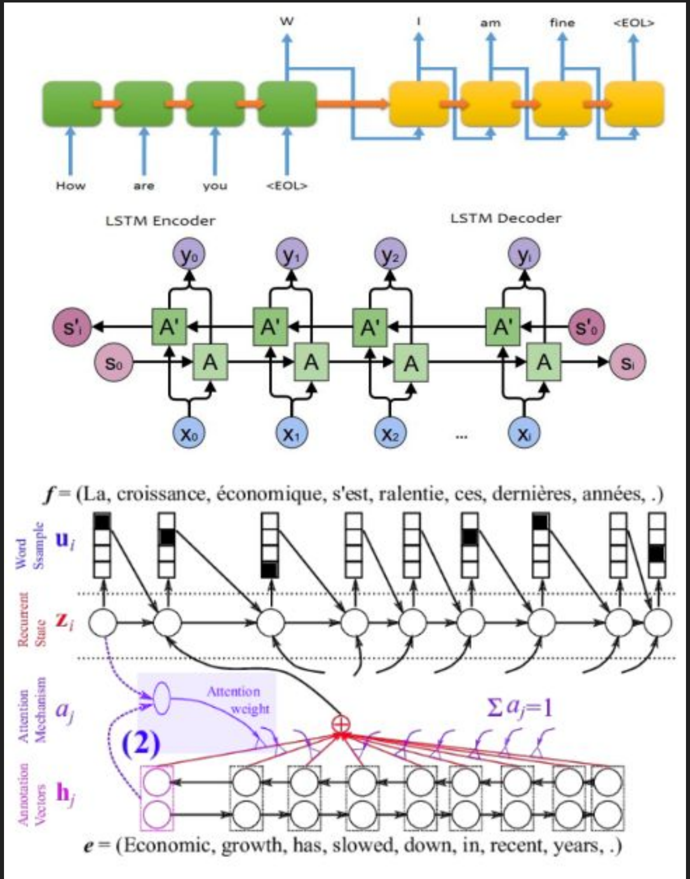
Text Analysis with Convolutional Neural Nets

What is currently used?

The “go-to” for Text analysis right now is Recurrent Neural Networks:

- Slow (er than CNNs)
- Cannot parallelize RNNs
- Complicated (if you want the whole sentence to be considered)
- Long term and backward dependencies imply almost double the computation
- Needs streamed sentences



Why use Convolutional Neural Nets for Text?

- Faster than RNNs
- Have a lot of knowledge and tools from the field
- Local invariance is useful for applications like Sentiment Analysis of text
- Compositionality of structures is inherent in grammar.
- And because why not?

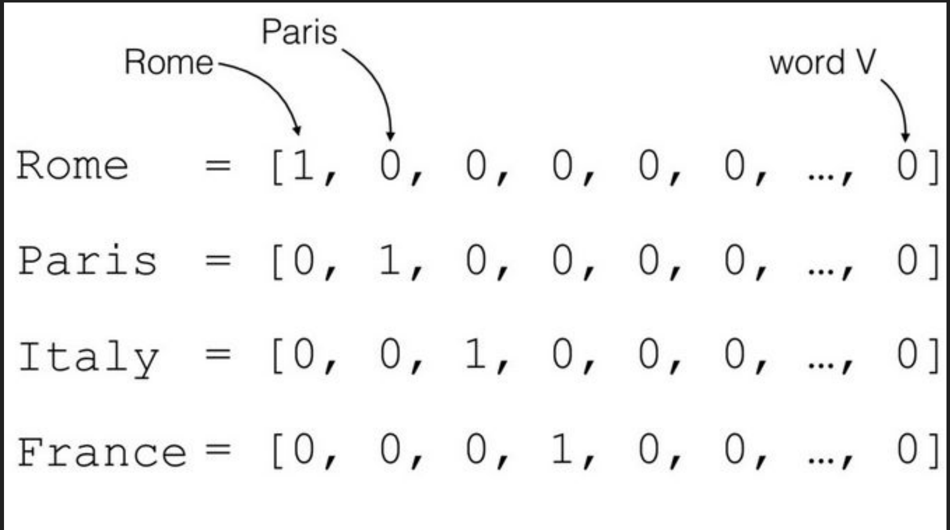
How do we convert some text into a 2D format?

We use some word embedding systems such as

- One-Hot Representations
- BoW : Bag of Words
- Word2Vec
- GloVe: Global Vectors

One Hot Representation Quickly Explained

- Trivial way of representing words
- Not always the most intuitive or helpful format
- Good as a base to start from
- Dimensions depend on the vocabulary.
- Doesn’t keep track of position of words



Word2Vec Quickly Explained

Properties of Word2Vec:

- Keep context of surrounding words
- Find common phrases in a vocabulary
- Creating a more compact representation of words and phrases.
- Embedding is created using **a simple feed forward neural net**

Achievements of word2vec:

- Encoding semantic properties of words into the representations
- Allowing for a mapping of relations in the corpus via a vector space of high dimensions.

Word2Vec Advantages

Allows for algebraic operations through the vector representations

w2v(King) = w2v(Queen) - w2v(woman) + w2v(man)

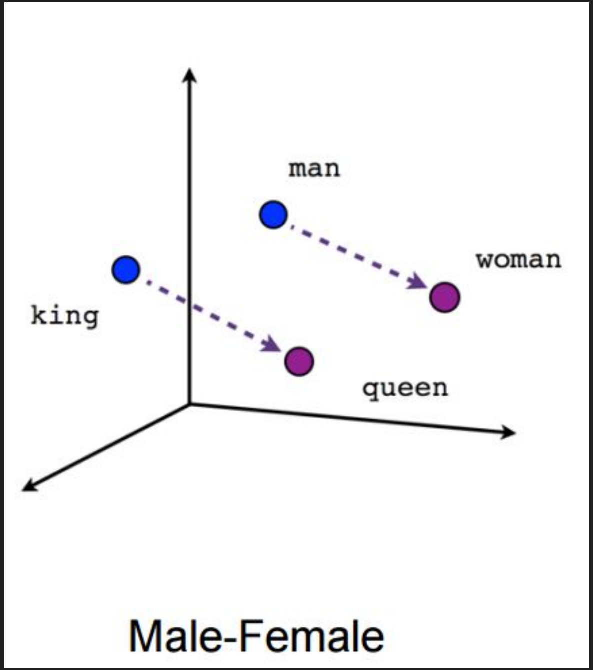
Is a valid formula.

Applications of Word2Vec:

- Semantic Relations
- Word Suggestions
- Language Translations

Aim/Applications of a Skip-Gram word2vec model:

Given Context, return the missing word!

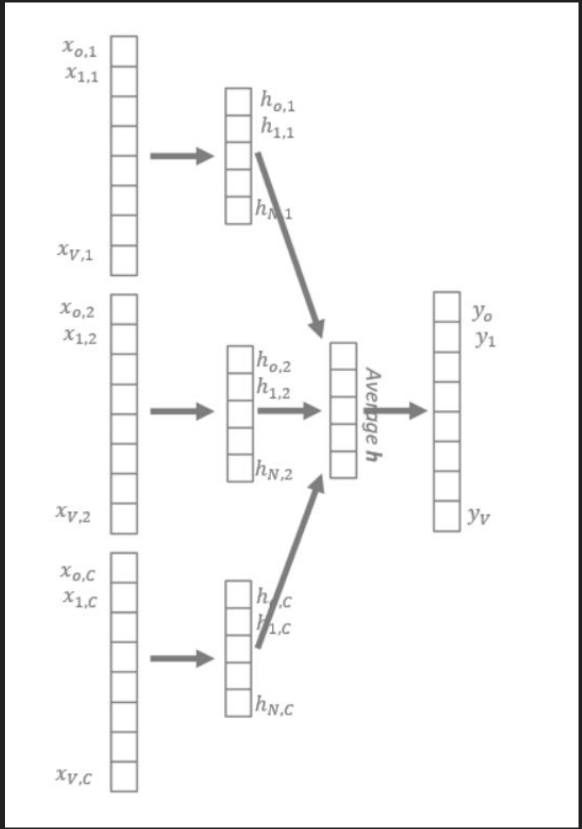


Word2Vec Quickly Explained: How it works? (Skip-Gram)

"I love baked potatoes"

Goal : Find the embedding of "baked"

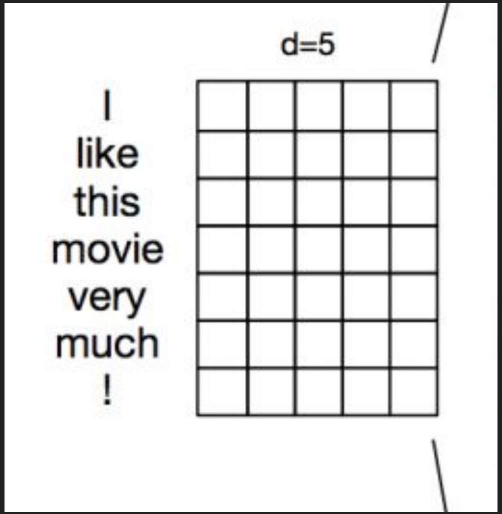
1. Run "i love potatoes" through a pretrained word2vec NN
2. Extract the one-hot representation of the word "baked"
3. Peel away the final layer of the network, and focus on the weights from the hidden layer
4. Weights in the hidden layer are used as the embedding



Embedding to 2D (word2vec)

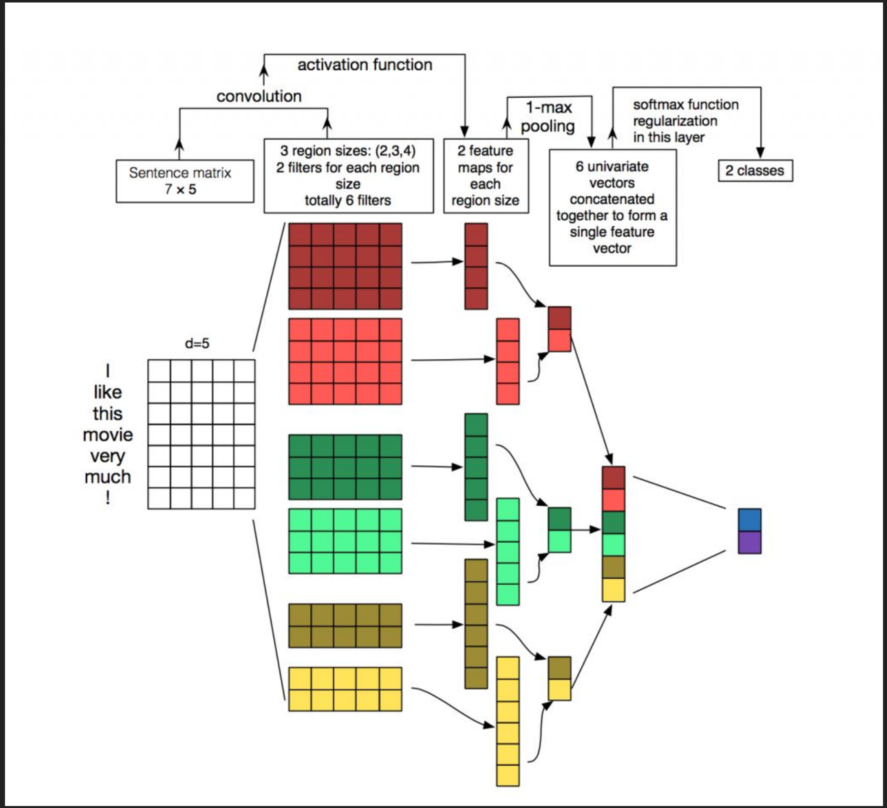
How to turn a sentence into 2D

1. Take the vector representation of each word in the sentence (d)
2. Take all the words in the sentence (n)
3. Create a matrix of size (n x d) which represents the sentence



Convolution on a text based matrix

- Convolutions are done over the whole width of the matrix
- Filters are of different sizes (n-gram distances)
- Max Pooling is often done over the whole feature map
- Softmax like functions bring down number of classes



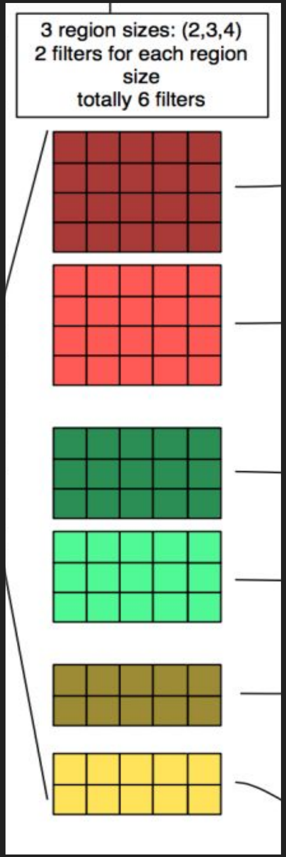
Text CNNs Filters

- Different sizes of filters
- Width = width of embedding
- Height = number of words wanted in the receptive field
- Capture different aspects of the sentence

For example:

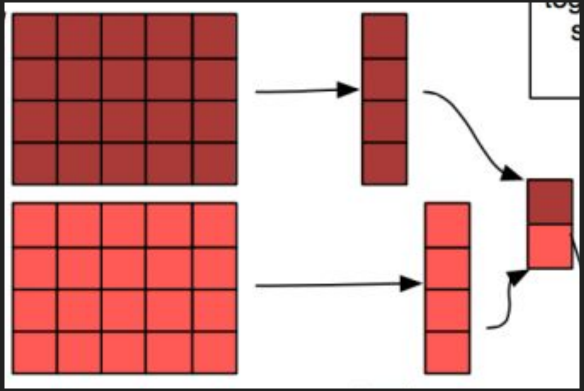
Size 2 filter: captures a negation “not amazing”

Size 4 filter: captures a target “potatoes are not amazing”



Text CNNs Pooling: The “not amazing” example

- Max Pooling is done after Convolutions
- Max Pooling Reduces dimensionality
- Obtains the core features of a sentence
- Max pooling is **done over the whole filter**



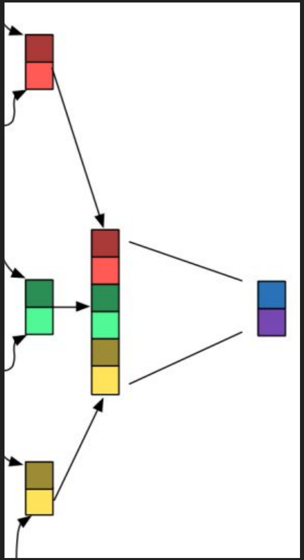
Negation Example:

In a sentence that contains “not amazing” as a phrase, we want to obtain the core idea that the sentence has a negation in it. We max pool to extract that information, rather than ‘noise’ in the rest of the sentence. (which is picked up by other filters)

Text CNNs Large Receptive Fields

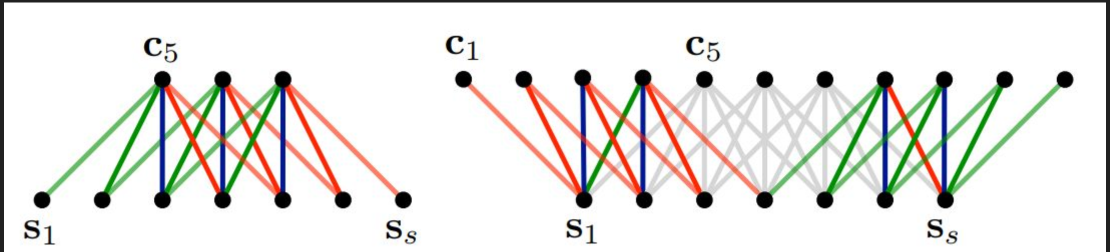
A great advantage of using CNNs is large receptive fields

- Various sizes of filters grab a large range of details
- Max pooling removes noise that exists in text (“is,the”)
- Allows the final layers to use information from each part of the sentence, and keep the context at the end influence the start of the sentence’s meaning.



Hyper Parameters of CNNs

- Various stride sizes allow us to mimic properties of a RNN, where we have large overlaps between contexts creating internal tree structures
- Wide / Narrow Convolutions: Allow us to alter how much of the embedding we want to use. Weighing certain relations and properties more than the others



Convolution on Images vs Text

Images

- Convolution helps local pixels
- Local Invariance is important (wherever the stop sign is in the image)
- RGB Channels in images
- Large receptive field to understand

Text

- Convolution helps nearby words
- Local invariance is important for sentiment / topic analysis
- Channels can be different embedding systems or languages