Lecture 8 - Natural Language Processing Fundamentals

Machine Learning Decal

Hosted by Machine Learning at Berkeley

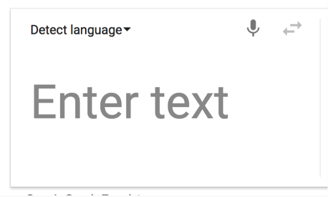
Agenda

How is NLP Used?

Text Classification

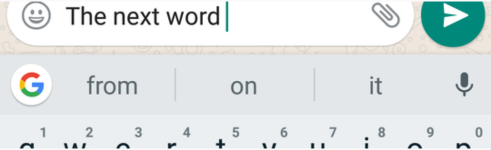
Categorize a document into a high level class using only the words in the document

- Sentiment analysis
- Language identification
- Spam filtering

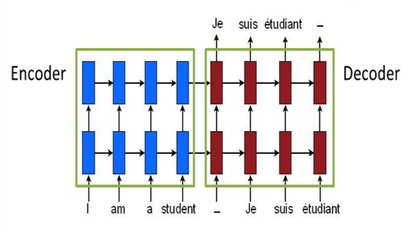


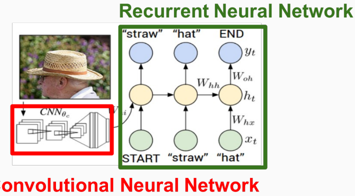
Language Modeling

- Predict a next word given previous words
- Encompasses many popular NLP tasks
- Makes most use of RNN/LSTMs



- Autocomplete a sentence
- Caption generation
- Machine translation





Word Embeddings

Areas of Study in NLP

- Syntax - grammatical structure
- Semantics - meaning of sentences

Each area of NLP has important contributions, but semantics is fundamental to understanding **what** a body of text is trying to convey.

Semantic Approaches

There are two fundamental ways to approach semantics:

- Propositional semantics - map a sentence to an expression in a logical language
    - "dog bites man" → bites(dog, man)
    - Meaning is attached to the function bites(*,*) where the first argument bites the second one
- Vector representation - map a sentence to a given vector where sentences similar to each other have similar embedding vectors

Propositional Semantics vs Vector Representation

Propositional Semantics

Vector Representation

Propositional Semantics vs Vector Representation

Propositional Semantics

- Allows for logical inferences

Vector Representation

Propositional Semantics vs Vector Representation

Propositional Semantics

- Allows for logical inferences
- More organized representation, but also difficult and expensive to define for every domain

Vector Representation

Propositional Semantics vs Vector Representation

Propositional Semantics

- Allows for logical inferences
- More organized representation, but also difficult and expensive to define for every domain

Vector Representation

- Appropriate vector representations can be learned → can be expanded to multiple domains

Propositional Semantics vs Vector Representation

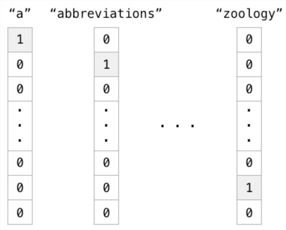
Propositional Semantics

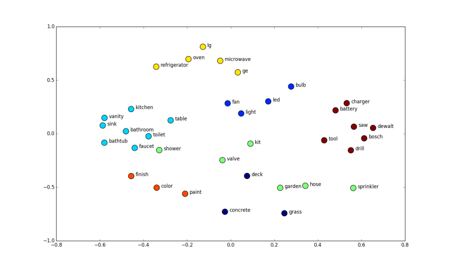
- Allows for logical inferences
- More organized representation, but also difficult and expensive to define for every domain

Vector Representation

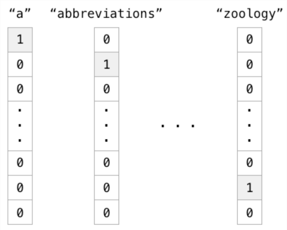
- Appropriate vector representations can be learned → can be expanded to multiple domains
- Good vector representation can allow basic analogies

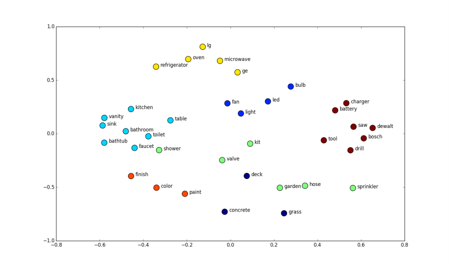
Sparse vs Dense Embeddings





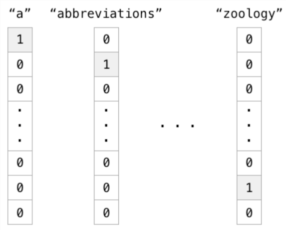
Sparse vs Dense Embeddings

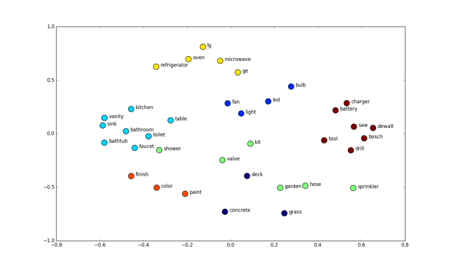




- Sparse embeddings take up more space and may not map similar words to similar vectors

Sparse vs Dense Embeddings





- Sparse embeddings take up more space and may not map similar words to similar vectors
- Properties of embeddings are very dependent on how they were generated

Properties of Good Word Embeddings

- Takes into account word similarity - similar words have similar embeddings

Key Idea: Similar words occur in similar contexts

- A good word embedding can be learned by using word contexts

Properties of Good Word Embeddings

- Takes into account word similarity - similar words have similar embeddings
- Efficient to store

Key Idea: Similar words occur in similar contexts

- A good word embedding can be learned by using word contexts

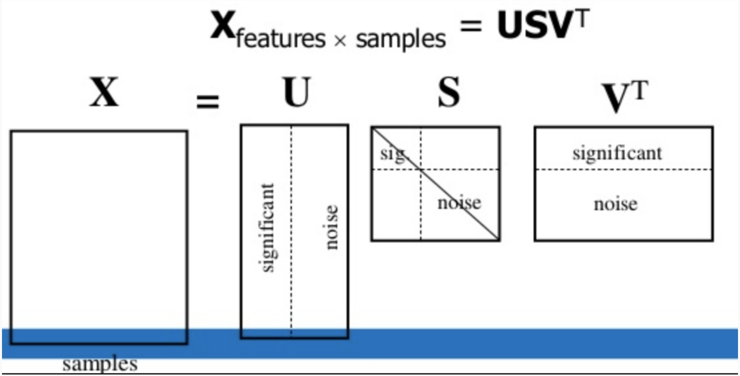
Properties of Good Word Embeddings

- Takes into account word similarity - similar words have similar embeddings
- Efficient to store
  - For reasonably sized vocabularies, one hot encoding is too expensive

Key Idea: Similar words occur in similar contexts

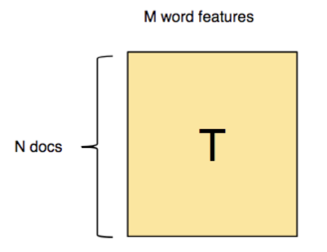
- A good word embedding can be learned by using word contexts

Recap: PCA


$\mathbf{X}_{\text{features} \times \text{samples}} = \mathbf{USV}^T$
X
=
U
S
V^T
significant
noise
sig
noise
significant
noise
samples

Latent Semantic Analysis

LSA is a document embedding method, similar to PCA, that uses word count matrices



$T_{i,j}$ is the count of word j in document i

Latent Semantic Analysis

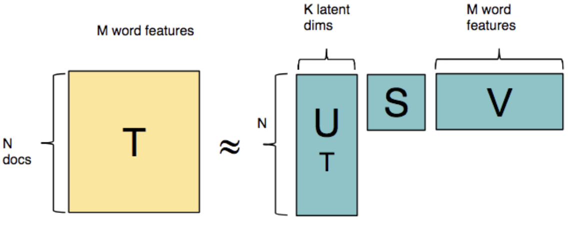
Word Count Matrix:

- I like deep learning.
- I like NLP.
- I enjoy flying.

	I	like	deep	learning	NLP	enjoy	flying
S1	1	1	1	1	0	0	0
S2	1	1	0	0	1	0	0
S3	1	0	0	0	0	1	1

This matrix T is called a bag of words matrix because ordering of the words is removed

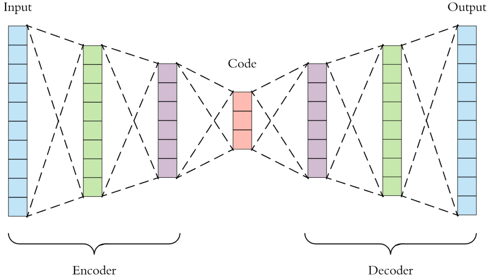
We can use an SVD decomposition on the bag of words matrix T to get document embeddings



If a new single document vector d is given with word counts, its document embedding is Vd

LSA as a Deep Network

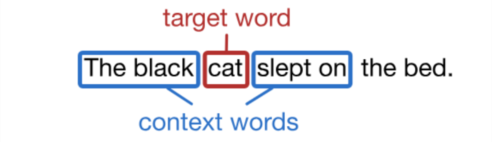
We know that SVD gives the best reconstruction T' of the bag of words matrix T when looking at the L2 error.



We can model this as an autoencoder trying to minimize the L2 reconstruction error.
Embedding given by passing document vector through the encoder.

Word2Vec - Local Contexts

- LSA uses entire document as a context for a single word - context region is too large
- Word2Vec only uses a few words surrounding a given word as context $\Rightarrow$ captures more nuanced semantic meaning



Word2Vec

There are 2 types of Word2Vec models

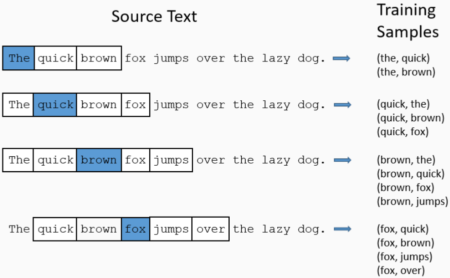
- Skip Gram - use the center word to predict the context words
- CBOW (Continuous Bag of Words) - use the context words to predict the center word

High Level Ideas:

- Train a single layer neural network for a task that we don't need solved
- Use the weights of that network to get rich embeddings of words

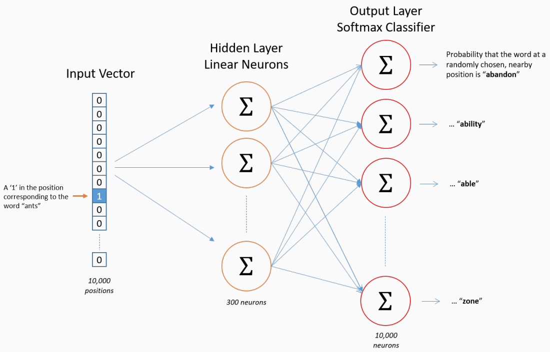
Word2Vec - Skip Gram

Goal: use the center word to predict the context words



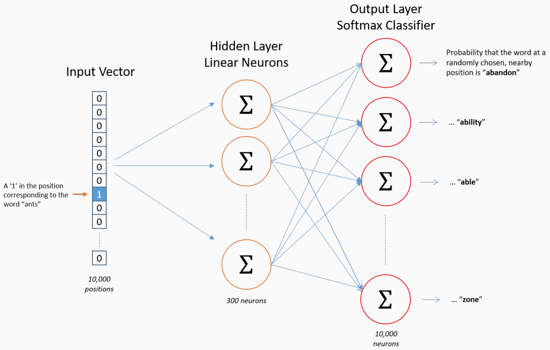
- For each center word, pick one context word at random for training
- For a particular center word, network will give probabilities of occuring in context for all vocabulary words

Word2Vec - Skip Gram



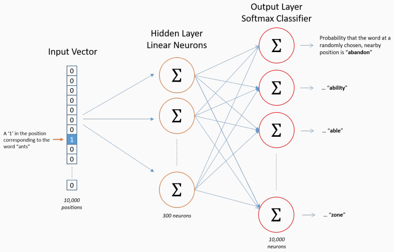
- For each center word, pick one context word at random for training
- For a particular center word, network will give probabilities of occuring in context for all vocabulary words

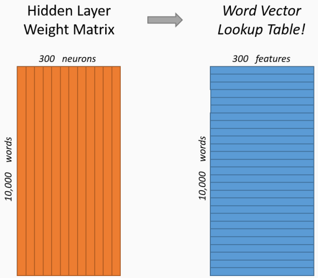
Word2Vec - Skip Gram



- Note that there are no nonlinear activations - this is so the weights of the network can directly be used as word embedding vectors

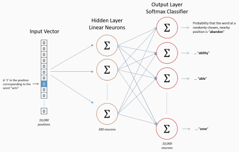
Word2Vec - Skip Gram

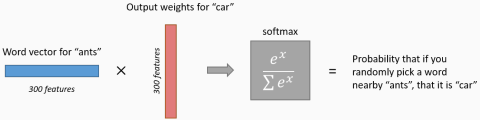




- Weight matrix for hidden layer has shape 10000x300 → this can directly become a lookup table for a 300 dimensional embedding for each word in the original vocabulary

Word2Vec - Skip Gram





- Suppose there is a high chance that context word $c1$ appears near center words $w1$ and $w2$
- Then the output probability of $c1$ from the skip gram network should be very similar regardless of whether $w1$ or $w2$ is the input
- The output weights for $c1$ is constant regardless of input word, so in order for output probabilities to be similar, both $w1$ and $w2$ should have similar word vectors

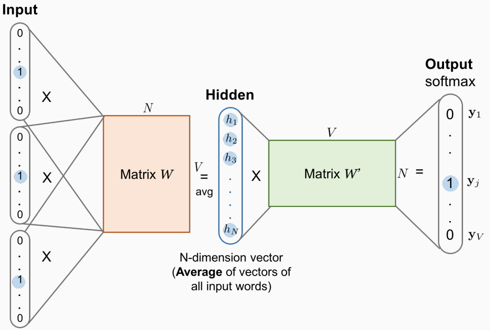
Word2Vec - Skip Gram

Interpretation of Skip Gram (and CBOW):

- Words with similar embeddings will likely be synonyms or very related because the contexts are similar
- Stems of words (eg: "ant" and "ants") also will have similar word vectors

Word2Vec - CBOW

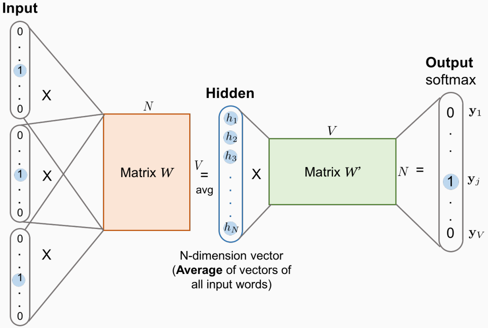
Goal: use the context words to predict the center word



- Each training example consists of all context words as input

Word2Vec - CBOW

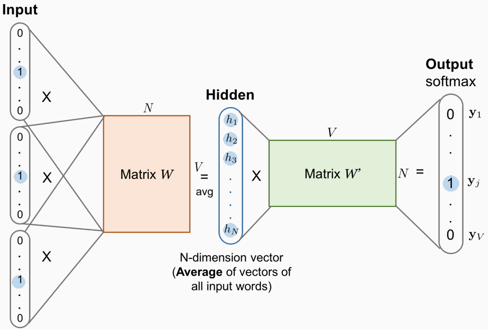
Goal: use the context words to predict the center word



- Model looks like inverted Skip Gram, but because there are multiple context words, after multiplying each context word by W, average the resulting vectors to get the hidden vector
- Note that no nonlinear activation functions are used again

Word2Vec - CBOW

Goal: use the context words to predict the center word



- ${W'}^T$ is used to get word embeddings

Word2Vec - CBOW vs Skip Gram

Skip Gram:

- Can better represent infrequent words
- Can train with relatively little training data

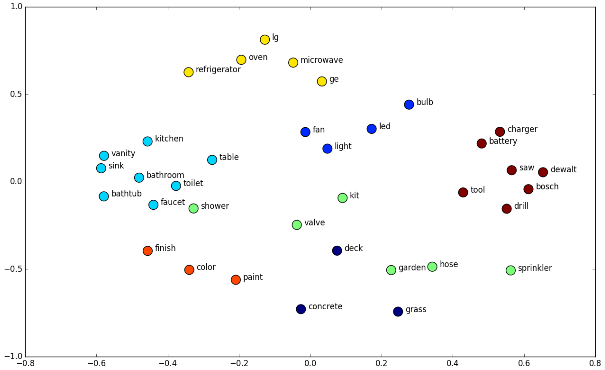
CBOW:

- Slightly better accuracy for frequent words
- Trains faster than Skip Gram

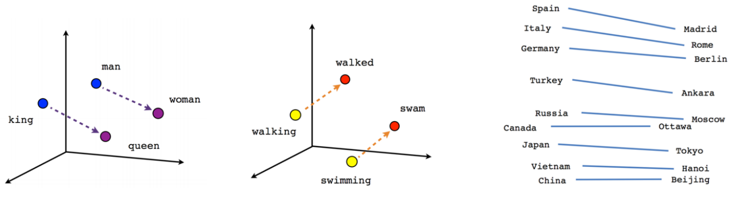
Some takeaways from Word2Vec

- Supervised learning, but "labels" are naturally occuring and plentiful
- Level of indirection - to get some information, another task is solved

Word2Vec - Learned Embeddings



Word2Vec - Learned Embeddings



Male-Female

Verb tense

Country-Capital

- Constant offsets between pair of words sharing particular relationship
- vec(man)-vec(woman) $\approx$ vec(king) - vec(queen)

Word2Vec - Learned Embeddings

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

GloVe

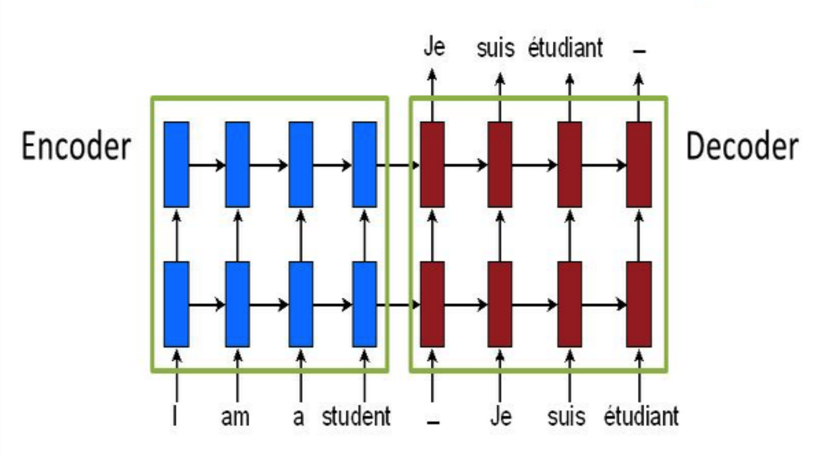
- Word2Vec is predictive model
  - It doesn't directly measure semantic distance (needed to create meaningful analogies)
  - Instead it learns vectors to help it accuratately predict target or context word
  - The task solved is different than why we want to use it
- GloVe is a count based model
  - Explicitly counts occurance of one word in context of other and learns vectors so the dot product of embeddings is equal probability of co-occurance
- Deep learning can be used to solve a similar task in different ways - possible to modify loss function or task overall

Transfer Learning with Word2Vec

- Learned word embeddings are analogous to the learned kernels in a CNN
- For a domain specific NLP task, can first train word vectors on a large general corpus and then update the Word2vec model using the domain specific data

Seq2Seq Networks for NLP

Vanilla Seq2Seq Network for Translation



Bleu Scores

Bleu scores are used to measure how close a generated translation is to the true translation.

Modified unigram precision: Clips counts by max occurances in reference sentence

$$\frac{\text{max correct number of unigrams in any reference sentence}}{\text{number unigrams in candidate sentence}}$$

- Candidate: **the the** the the the the the
- Reference 1: **the** cat is on **the** mat
- Reference 2: there is a cat on the mat

Modified unigram precision is $\frac{2}{7}$

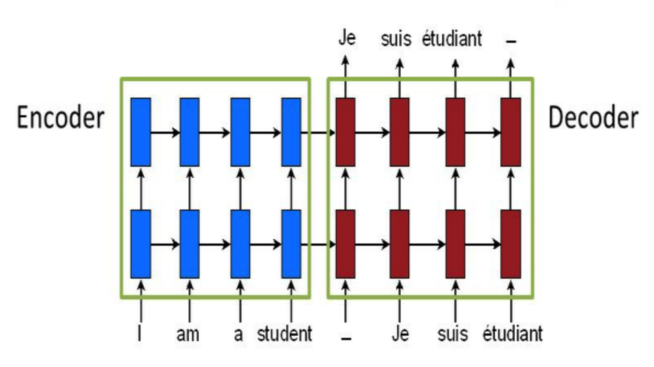
Bleu Scores

Can also use modified n-gram precision:

$$\frac{\text{max correct number of n-grams in any reference sentence}}{\text{number n-grams in candidate sentence}}$$

- Unigram precision scores capture adequecy of translation - does the translation contain the right words?
- N-gram precision scores capture fluency - how naturally does the translation read?

Issues with Vanilla Seq2Seq Networks



- All information from original sentence has to pass through bottleneck before decoding
- Bottleneck size is always fixed regardless of sentence length

Introduction to Attention

Attention

- Way of emphasizing certain information at different stages
  - In Seq2Seq translation models, we want the network to look at a certain part of the source sentence when translating that part
- One of the most important recent ideas in deep learning
- Can be used for computer vision, NLP, speech, RL

Soft vs Hard Attention

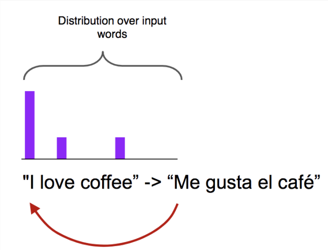
Hard Attention

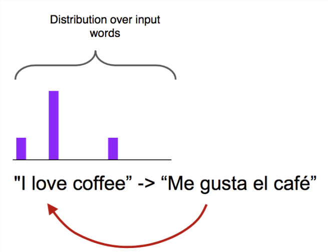
- Attention on only one input at each time step
- Cannot be trained with gradient descent - needs RL

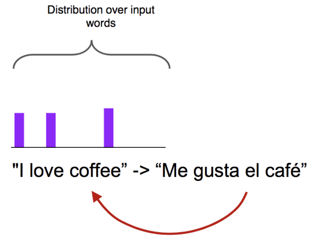
Soft Attention

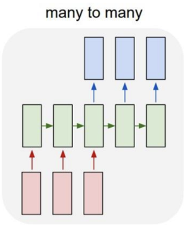
- Attention weighted across inputs

Soft Attention for Translation



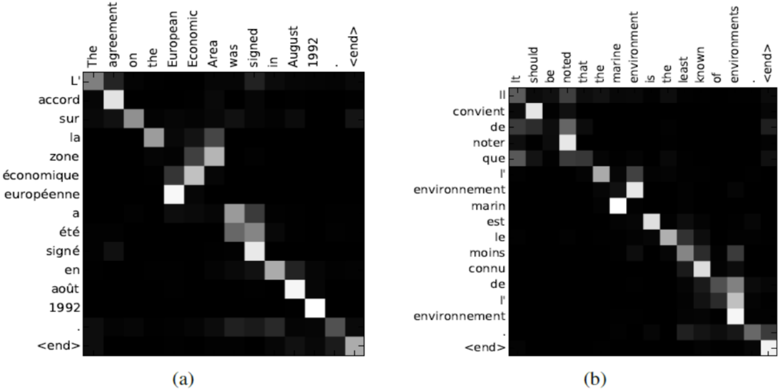






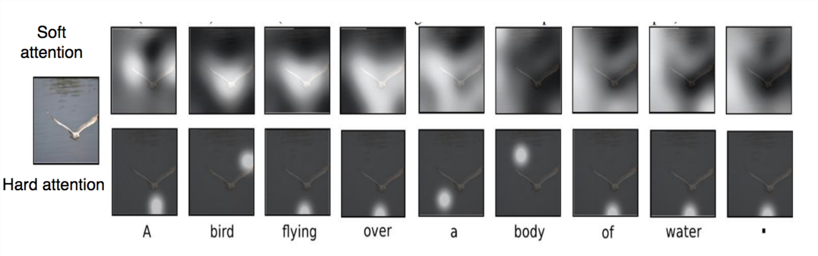
Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015

Soft Attention for Translation



Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015

Visualization of Attention - Image Captioning



Homework 6: Sentiment Analysis