Lexicon, Syntax, Semantics IIb: Modeling Meaning

Machine Learning for Meaning Representation

Eva Maria Vecchi

Center for Information and Language Processing
LMU Munich

July 9, 2020

Neural nets as machine learning algorithm

- NNs can be both supervised and unsupervised algorithms, depending on flavour:
  - multi-layer perceptron (MLP) – supervised
  - RNNs, LSTMs – supervised
  - auto-encoder – unsupervised
  - self-organising maps – unsupervised

Neural networks: a motivation

How to recognise digits?

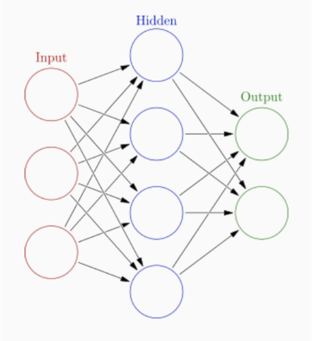
- Rule-based: a '1' is a vertical bar. A '2' is a curve to the right going down towards the left and finishing in a horizontal line…
- Feature-based: number of curves? of straight lines? directionality of the lines (horizontal, vertical)?
- Well, that's not gonna work…

Learning your own features

- We don't know what people pay attention to when recognizing digits (which features to use).
- Don't try to guess. Just let the system decide for you.
- A nice architecture to do this is the neural network:
  - Good for learning visual features.
  - Also good for learning latent linguistic features (remember SVD?)

Neural Nets

- A neural net is a set of interconnected neurons organised in 'layers'.
- Typically, we have one input layer, one output layer and a number of hidden layers in-between:

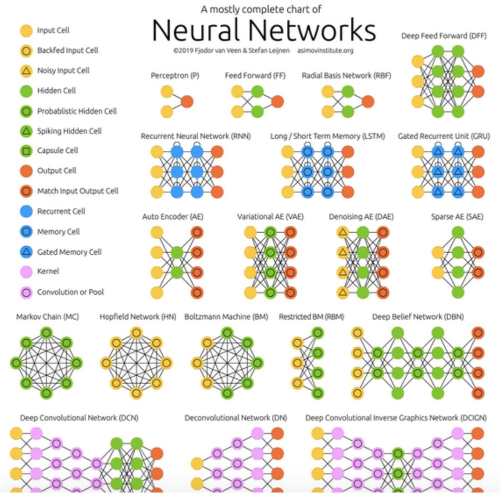


This is a multi-layer perceptron (MLP).

By Glosser.ca - Own work, Derivative of File:Artificial neural network.svg, CC BY-SA 3.0,

https://commons.wikimedia.org/w/index.php?curid=24913461

Neural network zoo



Go visit http://www.asimovinstitute.org/neural-network-zoo/ – very cool!

The artificial neuron

- The output of the neuron (also called 'node' or 'unit') is given by:

$$a = \varphi\left(\sum_{j=0}^{m} w_j x_j\right)$$

  where φ is the activation function.
- If this output is over a threshold, the neuron 'fires'.

A (simplified) example

- Should you bake a cake? It depends on the following features:
  - Wanting to eat cake (0/+1)
  - Having a new recipe to try (0/+1)
  - Having time to bake (0/+1)

(Herbelot 2018)

A (simplified) example

- Should you bake a cake? It depends on the following features:
  - Wanting to eat cake (0/+1)
  - Having a new recipe to try (0/+1)
  - Having time to bake (0/+1)
- How much weight should each feature have?
  - You like cake. Very much. *Weight: 0.8*
  - You need practice, as become a pastry chef is your professional plan B. *Weight: 0.3*
  - Baking a cake will take time away from your computational linguistics project, but you don't really care. *Weight: 0.1*

(Herbelot 2018)

A (simplified) example

- We'll ignore φ for now, so our equation for the output of the neuron is:

$$a = \sum_{j=0}^{m} w_j x_j$$

- Assuming you want to eat cake (+1), you have a new recipe (+1) and you don't really have time (0), our output is:

$$0.8 * 1 + 0.3 * 1 + 0.1 * 0 = 1.1$$

- Let's say our threshold is 0.5, then the neuron will fire (output 1). You should definitely bake a cake.

(Herbelot 2018)

From threshold to bias

- We can write $\sum_{j=0}^{m} w_j x_j$ as the dot product $\overrightarrow{w} \cdot \overrightarrow{x}$
- We usually talk about bias rather than threshold – which is just a way to move the value to the other side of our inequality:
  - if $\overrightarrow{w} \cdot \overrightarrow{x} > t$, then 1 (fire) else 0
  - if $\overrightarrow{w} \cdot \overrightarrow{x} - t > 0$, then 1 (fire) else 0
- The bias is a 'special neuron' in each layer, with a connection to all other units in that layer.

But hang on. . .

- Didn't we say we didn't want to encode features? Those inputs look like features. . .
- Right. In reality, what we will be inputting are not human-selected features but simply a vectorial representation of our input.
- Typically, we have one neuron per value in the vector.
- Similarly, we have a vectorial representation of our output (which could be as simple as a single neuron representing a binary decision).

Neural nets and word meaning

Instead of creating the co-occurrence matrix and then reducing its dimensions...

Why not learn the compressed representations directly from the data?

Word embeddings

- give words from a vocabulary as input to a (feed-forward) neural network
- embed them as vectors into a lower dimension space
- fine-tune through back-propagation
- $\longrightarrow$ yields word embeddings as the weights of the first layer, usually referred to as *Embedding Layer*
- The objective is to create word representations (embeddings) that are good at predicting the surrounding context.

Distributional vs. Distributed Representation

Distributional Representation

- captures linguistic distribution of each word in form of a high-dimensional numeric vector
- typically based on co-occurrence counts (aka "count" models)
- based on distributional hypothesis: similar distribution ˜similar meaning (similar distribution = similar representation)

Distributional vs. Distributed Representation

Distributional Representation

- captures linguistic distribution of each word in form of a high-dimensional numeric vector
- typically based on co-occurrence counts (aka "count" models)
- based on distributional hypothesis: similar distribution s̃imilar meaning (similar distribution = similar representation)

Distributed Representation

- sub-symbolic, compact representation of words as dense numeric vector
- meaning is captured in different dimensions and it is used to predict words (aka "predict" models)
- similarity of vectors corresponds to similarity of the words
- aka word embeddings

Methods to train word embeddings

- First and most used: word2vec (see below)
- FastText: similar to word2vec but trained on character n-grams instead of words
- GloVe: Global Vectors - first uses co-occurrence matrix, calculates ratios of probabilities; trained with log-bilinear regression model
- ELMo, BERT, Flair: Contextualized word embeddings
- among many others. . .

word2vec

Mikolov et al, 2013

Framework for learning word embeddings; main idea:

- takes words from a very large corpus of text as input (unsupervised)
- learn a vector representation for each word to predict between every word and its context
- fully connected feed-forward neural network with one hidden layer
- Two main algorithms:
  - **Continuous Bag of Words (CBOW)**: predicts center word from the given context (sum of surrounding words vectors)
  - **Skip-gram**: predicts context taking the center word as input

Center word and context

- Embedding models consider the history (previous words) **and** the future (following words) of a center word[1]
- The number of words considered is called the window size (standard size = 5)
- **Importance of window size**
  - "Australian scientists discover stars with telescopes."
  - context window size 2 center word context window size 2
  - Note: different meaning of "stars" with and without telescope

[1]Unlike language models, who only look at past words for predictions

CBOW

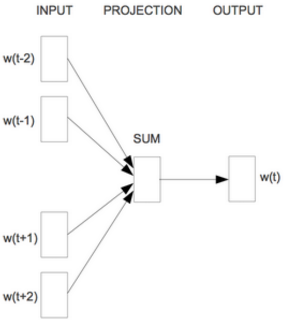


Figure 4: Continuous bag-of-words (Mikolov et al., 2013)

- It uses continuous representations whose order is of no importance
- CBOW can be seen as a precognitive language model
- Objective function similar to a language model

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} logp(w_t : | : w_{t-n}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+n})$$

Skip-gram

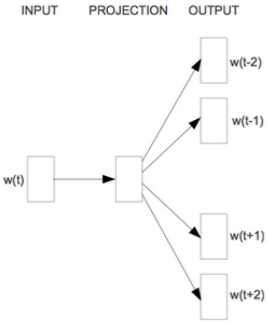


Figure 5: Skip-gram (Mikolov et al., 2013)

- Instead of using the surrounding words to predict the centre word as with CBOW, skip-gram uses the centre word to predict the surrounding words
- objective thus sums the log probabilities of the surrounding n words to the left and to the right of the target word w_t

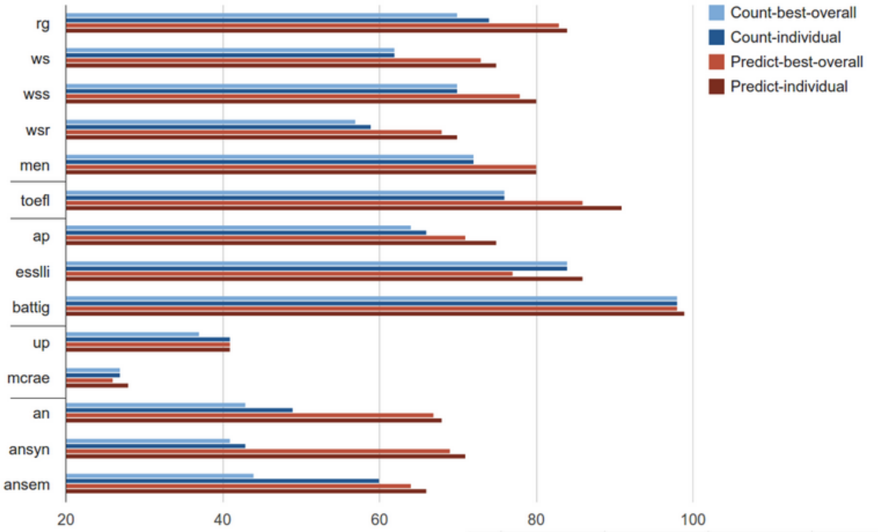
$$J_\theta = \frac{1}{T}\sum_{t=1}^{T}\sum_{-n\leq j\leq n,\neq 0} logp(w_{t+j} : | : w_t)$$

Don't count, predict!

Baroni et al., ACL 2014

Don't count, predict!

Baroni et al., ACL 2014



Evaluation: Analogy

$$\overrightarrow{king} - \overrightarrow{man} + \overrightarrow{woman}$$

Evaluation: Analogy

$$\overrightarrow{king} - \overrightarrow{man} + \overrightarrow{woman}$$

1. queen
2. monarch
3. princess
4. kings ...

- Word Analogy Task: a is to b, as c is to *?*
- How many word analogies can the trained embeddings predict correctly?

Evaluation: Topics

$$\overrightarrow{hungry} + \overrightarrow{monster}$$

Evaluation: Topics

$$\overrightarrow{hungry} + \overrightarrow{monster}$$

- monsters, beast, ravenous, creature, monstrous, starving, famished, hunger, thirsty, cannibal, ravening, ...
- Pretty good, but a lot of emphasis on (food-like) hunger

Evaluation: Topics

$$\overrightarrow{hungry} + \overrightarrow{monster}$$

- monsters, beast, ravenous, creature, monstrous, starving, famished, hunger, thirsty, cannibal, ravening, ...
- Pretty good, but a lot of emphasis on (food-like) hunger

$$\overrightarrow{hungry} + \overrightarrow{monster} - \overrightarrow{food}$$

- Refine topic by removing too generically food-related words
- monstrous, monsters, beast, ravenous, ogre, monster-like, three-headed, child-eating, creature, mad, bloodthirsty, frightened, ...

Evaluation: Descriptions

2. Choose the most appropriate **verb** from the list in the boxes for each of the situations below. Use each of the words only **once**.

sidle	**amble**	**totter**	**trudge**
stagger	**strut**	**march**	**lurch**

a) A baby just learning to walk totter

b) A drunk man walking down the street lurch

c) A weary farmer returning home through the mud trudge

d) Two teenagers guiltily approaching someone sidle

e) Someone who has just been shot Stagger

f) A lazy walk in the country amble

g) A model walking down the catwalk Strut

h) Someone going to the manager of a hotel to make a strong complaint march

Choose the most appropriate word for each space from the bracketed words:

Evaluation: Descriptions

- **baby learning to walk**
  - amble, totter, trudge, stagger, march, strut, sidle, lurch
- **drunk man walking down street**
  - stagger, amble, trudge, march, trotter, sidle, strut, lurch
- **weary farmer returning home through mud**
  - trudge, stagger, amble, lurch, march, totter, strut, sidle
- **two teenagers guiltily approaching someone**
  - sidle, amble, stagger, totter, trudge, lurch, march, strut
- **someone who has just been shot**
  - stagger, march, trudge, sidle, lurch, amble, strut, totter
- **a lazy walk in the country**
  - amble, trudge, stagger, march, totter, sidle, strut, lurch

The unreasonable effectiveness ...

- In 2015, Andrej Karpathy wrote a blog entry which became famous: *The unreasonable effectiveness of Recurrent Neural Networks*[2].

- How a simple model can be unbelievably effective.

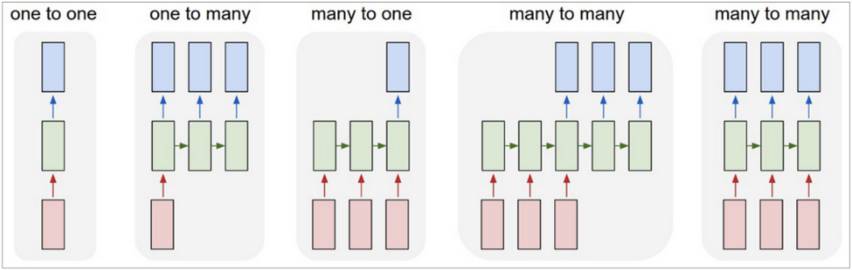
[2]https://karpathy.github.io/2015/05/21/rnn-effectiveness/

Recurrence

- Feedforward NNs which take a vector as input and produce a vector as output are limited.
- Putting recurrence into our model, we can now process *sequences* of vectors, at each layer of the network.

Architectures

What might these architectures be used for?



https://karpathy.github.io/2015/05/21/rnn-effectiveness/

Language Modeling

- A language model (LM) is a model that computes the probability of a sequence of words, given some previously observed data.
- LMs are used widely, for instance in predictive text on your smartphone:

Today, I am in (my/bed/Munich/Ulaanbaatar).

(Herbelot 2018)

The Markov assumption

- Let's assume the following sentence:

I am in Rome.

- We are going to use the chain rule for calculating its probability:

$$P(A_n, \ldots, A_1) = P(A_n | A_{n-1}, \ldots, A_1) \cdot P(A_{n-1}, \ldots, A_1)$$

- For our example,

$$P(I, am, in, Rome) = P(Rome|in, am, I) \cdot P(in|am, I) \cdot P(am|I) \cdot P(I)$$

(Herbelot 2018)

The Markov assumption

- The problem is, we cannot easily estimate the probability of a word in a long sequence.
- There are too many possible sequences that are not observable in our data or have very low frequency:

$$P(Rome|in, am, I, today, but, yesterday, there, \ldots)$$

- So we make a simplifying Markov assumption:

$$P(Rome|in, am, I) \approx P(Rome|in)(bigram)$$

or

$$P(Rome|in, am, I) \approx P(Rome|in, am)(trigram)$$

- That is, we are not taking into account *long-distance dependencies* in language.
- Trade-off between accuracy of the model and trainability.

(Herbelot 2018)

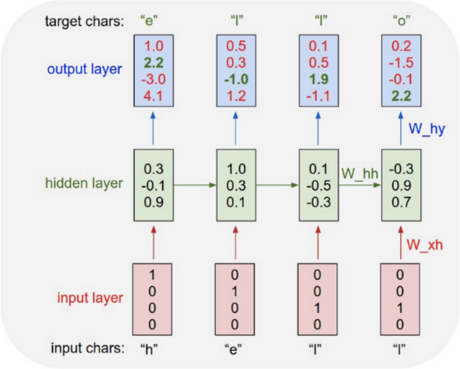
LMs as a generative model

- In your smartphone, the LM does not just calculate a sentence probability, it suggests the next word to what you're writing.
- Given the sequence *I am in*, for each word w in the vocabulary, the LM can calculate:

$$P(w|in, am, I)$$

- The word with the highest probability is returned

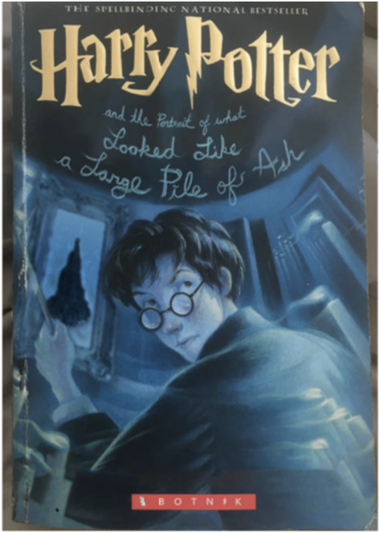
Language modeling with RNNs

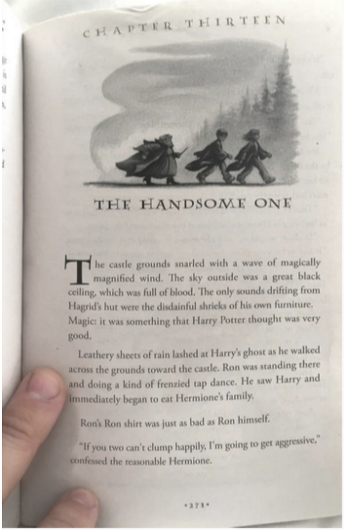


- The *sequence* given to the RNN is equivalent to the n-gram of a language model.
- Given a word or character, it has to predict the next one.

(Herbelot 2018)

Example: Rewriting Harry Potter





http://www.botnik.org/content/harry-potter.html

Types of recurrent NNs

- **RNNs (Recurrent Neural Networks)**: the original version. Simple architecture but does not have much memory.
- **LSTMs (Long Short-Term Memory Networks)**: an RNN able to remember and forget selectively.
- **GRUs (Gated Recurrent Units)**: a variation on LSTMs.

Trends and future directions

Subword-level embeddings

- Word embeddings have been augmented with subword-level information for many applications
  - e.g. named entity recognition, part-of-speech tagging, dependency parsing, and language modelling
- Often use a CNN or a BiLSTM
  - input: characters of a word
  - output: a character-based word representation
- Character n-grams features have been shown to be more powerful than composition functions over individual characters
- Even smaller!
  - subword units based on *byte-pair encoding* perform well for machine translation and entity typing
  - easily learned, but no real advantage over character-based representations for most tasks

OOV handling

- Main problem with pre-trained embeddings: unable to deal with out-of-vocabulary (OOV) words
- One solution: subword-level embeddings – some success
- Recent approaches aim to generate OOV embedding on-the-fly
  - e.g. initialize the embedding of OOV words as the sum of their context words, then rapidly refine only the OOV embedding with a high learning rate (Herbelot & Baroni, 2017)

Evaluation

- Evaluation of pre-trained embeddings remain a contentious issue
  - word similarity and analogy datasets have been shown to only correlate weakly with downstream performance
- The RepEval Workshop at ACL 2016 exclusively focused on better ways to evaluate pre-trained embeddings
- So far, best way to evaluate: extrinsic evaluation on downstream tasks

Multi-sense embeddings

- Common criticism: embeddings are unable to capture polysemy
- Most approaches for learning multi-sense embeddings solely evaluate on word similarity
- *However*, strong results in Neural Machine Translation $\longrightarrow$ models are expressive enough to contextualize and disambiguate words in context
- Yet, still need to understand if and how models are disambiguating, and how to improve if necessary

Phrases and multi-word expressions

- Problem: Embeddings fail to capture the meanings of phrases and multi-word expressions
  - Eg. *kick the bucket*, *work hard*, *take a seat*, etc.
- Some attempts to build phrase embeddings or better learn compositional and non-compositional phrases
- explicitly modelling phrases has so far not shown significant improvements on downstream tasks that would justify the additional complexity
- a better understanding of how phrases are modelled in neural networks would allow methods that augment the capabilities of our models to capture compositionality and non-compositionality of expressions

S. Ruder, *Word embeddings in 2017: Trends and future directions*

Bias

- Word embeddings trained on, e.g., Google News articles exhibit female/male gender stereotypes to a disturbing extent (Bolukbasi et al., 2016)
- Bias in models becoming big issue in the field
- What other biases are captured in embeddings, and how best to remove bias?

Temporal dimension

- Word meanings are subject to continuous change
- We can consider the temporal dimension and the diachronic nature of words
- Useful to reveal laws of semantic change, model temporal word analogy or relatedness, and capture dynamics of semantic relations.

Lack of theoretical understanding

- Little work on gaining a better theoretical understanding of the word embedding space and its properties
    - e.g. that summation captures analogy relations
- Some insights explored in Arora et al. (2016), Gittens et al. (2017), Mimno & Thompson (2017)

Task and domain-specific embeddings

- Downside of pre-trained embeddings: news data for training often different than data for tasks
    - also hard to come by millions of unlabelled docs in most target domains
- Some attempts to adapt pre-trained embeddings to capture characteristics of target domain, and retain relevant existing knowledge
    - e.g. Lu & Zheng (2017): regularized skip-gram model to learn cross-domain embeddings
- Or, use existing knowledge from semantic lexicons to augment pre-trained embeddings with relevant information
    - e.g. retro-fitting (Faruqui et al., 2015), injecting prior knowledge like monotonicity (You et al., 2017) and word similarity (Niebler et al., 2017), etc.

S. Ruder, *Word embeddings in 2017: Trends and future directions*

Transfer learning

- Aim to create *contextualized* word vectors (rather than adapting them)
  - augment word embeddings with embeddings based on hidden states of models pre-trained for certain tasks
  - e.g. machine translation or language modeling
- **Bidirectional Encoder Representations from Transformers (BERT)**
  - introduced by Google AI in 2018
  - first deeply *bidirectional, unsupervised* language representation, pre-trained using only a plain text corpus
  - stunning results on numerous NLP tasks

Embeddings for multiple languages

- Goal to create multilingual word embeddings
- Methods being developed that learn cross-lingual representations with as few parallel data as possible
- Some work also aims to learn multilingual embeddings *without* parallel data
- Note issue of training for low-resource languages

S. Ruder, *Word embeddings in 2017: Trends and future directions*

Thanks! Any questions?

`https://www.vecchi.com/eva/teaching/modelingmeaning.html`