Compositional Distributional Models of Meaning

Dimitri Kartsaklis Mehrnoosh Sadrzadeh

School of Electronic Engineering
and Computer Science

Queen Mary
University of London

COLING 2016

11th December 2016
Osaka, Japan

In a nutshell

- Compositional distributional models of meaning (CDMs) extend distributional semantics to the phrase/sentence level.
- They provide a function that produces a vectorial representation of the meaning of a phrase or a sentence from the distributional vectors of its words.
- Useful in every NLP task: sentence similarity, paraphrase detection, sentiment analysis, machine translation etc.

In this tutorial:

We review three generic classes of CDMs: vector mixtures, tensor-based models and neural models.

Outline

1. Introduction
2. Vector Mixture Models
3. Tensor-based Models
4. Neural Models
5. Afterword

Computers and meaning

- How can we define Computational Linguistics?

  Computational linguistics is the scientific and engineering discipline concerned with understanding written and spoken language from a computational perspective.

  —Stanford Encyclopedia of Philosophy[1]

[1] http://plato.stanford.edu

Compositional semantics

The principle of compositionality

The meaning of a complex expression is determined by the meanings of its parts and the rules used for combining them.

- Montague Grammar: A systematic way of processing fragments of the English language in order to get semantic representations capturing their meaning.

> *There is in my opinion no important theoretical difference between natural languages and the artificial languages of logicians.*
>
> —Richard Montague, *Universal Grammar* (1970)

Syntax-to-semantics correspondence (1/2)

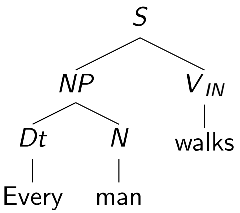
- A lexicon:

(1) a. every $\vdash Dt : \lambda P.\lambda Q.\forall x[P(x) \rightarrow Q(x)]$

b. man $\vdash N : \lambda y.man(y)$

c. walks $\vdash V_I : \lambda z.walk(z)$

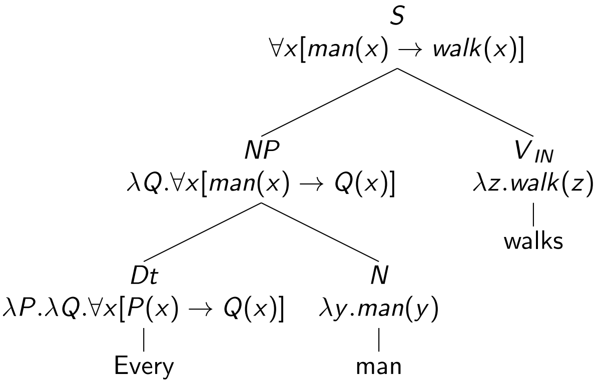
- A parse tree, so syntax guides the semantic composition:



$$NP \rightarrow Dt \;\; N : [\![N]\!]([\![Dt]\!])$$
$$S \rightarrow NP \;\; V_{IN} : [\![V_{IN}]\!]([\![NP]\!])$$

Syntax-to-semantics correspondence (2/2)

- Logical forms of compounds are computed via β-reduction:



- The semantic value of a sentence can be **true** or **false**.
- Can we do better than that?

The meaning of words

Distributional hypothesis

Words that occur in similar contexts have similar meanings [Harris, 1958].

The functional interplay of philosophy and	?	should, as a minimum, guarantee...
...and among works of dystopian	?	fiction...
The rapid advance in	?	today suggests...
...calculus, which are more popular in	?	-oriented schools.
But because	?	is based on mathematics...
...the value of opinions formed in	?	as well as in the religions...
...if	?	can discover the laws of human nature....
...is an art, not an exact	?	.
...factors shaping the future of our civilization:	?	and religion.
...certainty which every new discovery in	?	either replaces or reshapes.
...if the new technology of computer	?	is to grow significantly
He got a	?	scholarship to Yale.
...frightened by the powers of destruction	?	has given...
...but there is also specialization in	?	and technology...

The meaning of words

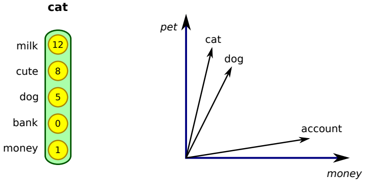
Distributional hypothesis

Words that occur in similar contexts have similar meanings [Harris, 1958].

The functional interplay of philosophy and	**science**	should, as a minimum, guarantee...
...and among works of dystopian	**science**	fiction...
The rapid advance in	**science**	today suggests...
...calculus, which are more popular in	**science**	-oriented schools.
But because	**science**	is based on mathematics...
...the value of opinions formed in	**science**	as well as in the religions...
...if	**science**	can discover the laws of human nature....
...is an art, not an exact	**science**	.
...factors shaping the future of our civilization:	**science**	and religion.
...certainty which every new discovery in	**science**	either replaces or reshapes.
...if the new technology of computer	**science**	is to grow significantly
He got a	**science**	scholarship to Yale.
...frightened by the powers of destruction	**science**	has given...
...but there is also specialization in	**science**	and technology...

Distributional models of meaning

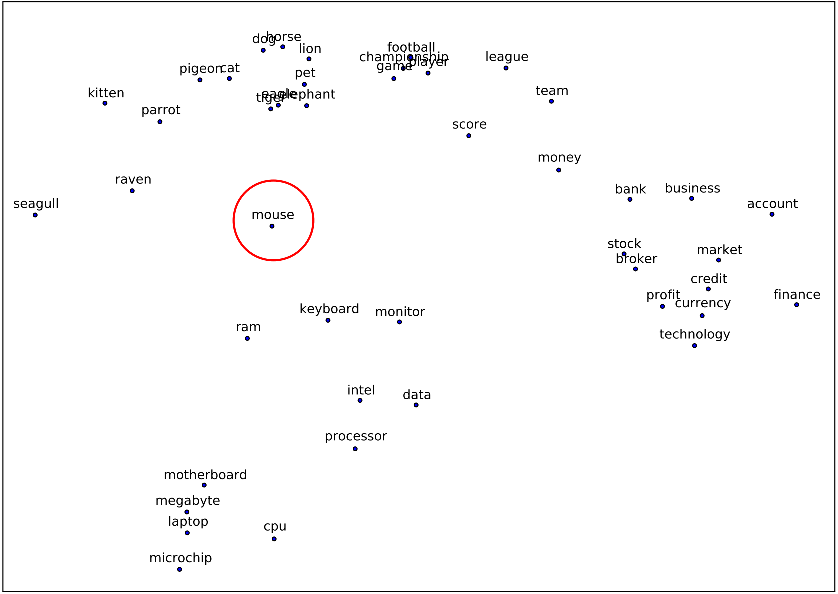
- A word is a vector of co-occurrence statistics with every other word in a selected subset of the vocabulary:



- Semantic relatedness is usually based on cosine similarity:

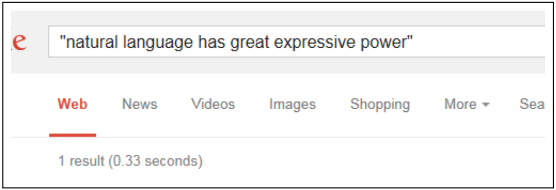
$$\text{sim}(\overrightarrow{v}, \overrightarrow{u}) = \cos\theta_{\overrightarrow{v},\overrightarrow{u}} = \frac{\langle \overrightarrow{v} \cdot \overrightarrow{u} \rangle}{\|\overrightarrow{v}\|\|\overrightarrow{u}\|}$$

A real vector space



The necessity for a unified model

- Distributional models of meaning are quantitative, but they do not scale up to phrases and sentences; there is not enough data:



> Even if we had an infinitely large corpus,
> what the context of a sentence would be?

The role of compositionality

Compositional distributional models

We can produce a sentence vector by composing the vectors of the words in that sentence.

$$\overrightarrow{s} = f(\overrightarrow{w_1}, \overrightarrow{w_2}, \ldots, \overrightarrow{w_n})$$

Three generic classes of CDMs:

- *Vector mixture* models [Mitchell and Lapata (2010)]
- *Tensor-based* models [Coecke, Sadrzadeh, Clark (2010); Baroni and Zamparelli (2010)]
- *Neural* models [Socher et al. (2012); Kalchbrenner et al. (2014)]

A CDMs hierarchy

Theoretical power		Distinguishing feature	Linguistic justification	Space complexity	Training time
Non-linear	Deep learning models	Drastic transformation of sentence space geometry	Med	Med	High
Multi-linear	Tensor-based models	Relational words are functions applied on argument vectors	High	High	Med
Element-wise mixing	Vector mixture models	Equal contribution of all words	Low	Low	—

Applications (1/2)

Why CDMs are important?

The problem of producing robust representations for the meaning of phrases and sentences is at the heart of every task related to natural language.

- **Paraphrase detection**

  Problem: Given two sentences, decide if they say the same thing in different words

  Solution: Measure the cosine similarity between the sentence vectors

- **Sentiment analysis**

  Problem: Extract the general sentiment from a sentence or a document

  Solution: Train a classifier using sentence vectors as input

Applications (2/2)

- **Textual entailment**

  Problem: Decide if one sentence logically infers a different one

  Solution: Examine the feature inclusion properties of the sentence vectors

- **Machine translation**

  Problem: Automatically translate one sentence into a different language

  Solution: Encode the source sentence into a vector, then use this vector to decode a surface form into the target language

- *And so on. Many other potential applications exist...*

Outline

1. Introduction
2. Vector Mixture Models
3. Tensor-based Models
4. Neural Models
5. Afterword

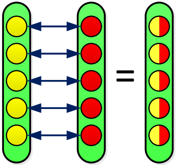
Element-wise vector composition

- The easiest way to compose two vectors is by working element-wise [Mitchell and Lapata (2010)]:

$$\overrightarrow{w_1 w_2} = \alpha \overrightarrow{w_1} + \beta \overrightarrow{w_2} = \sum_i (\alpha c_i^{w_1} + \beta c_i^{w_2}) \overrightarrow{n_i}$$

$$\overrightarrow{w_1 w_2} = \overrightarrow{w_1} \odot \overrightarrow{w_2} = \sum_i c_i^{w_1} c_i^{w_2} \overrightarrow{n_i}$$

- An element-wise "mixture" of the input elements:



Properties of vector mixture models

- Words, phrases and sentences share the same vector space
- A bag-of-word approach. Word order does not play a role:

$$\overrightarrow{dog} + \overrightarrow{bites} + \overrightarrow{man} = \overrightarrow{man} + \overrightarrow{bites} + \overrightarrow{dog}$$

- Feature-wise, vector addition can be seen as feature union, and vector multiplication as feature intersection

Vector mixtures: Intuition

- The distributional vector of a word shows the extent to which this word is related to other words of the vocabulary
- For a verb, the components of its vector are related to the action described by the verb
- I.e. the vector for the word 'run' shows the extent to which a 'dog' can run, a 'car' can run, a 'table' can run and so on
- So, the element-wise composition of $\overrightarrow{dog}$ with $\overrightarrow{run}$ shows the extent to which things that are related to dogs can also run (and vice versa); in other words:

> The resulting vector shows how compatible is the verb with the specific subject.

Vector mixtures: Pros and Cons

Distinguishing feature:

All words contribute equally to the final result.

PROS:

- Trivial to implement
- Surprisingly effective in practice

CONS:

- A bag-of-word approach
- Does not distinguish between the type-logical identities of the words

Outline

1 Introduction

2 Vector Mixture Models

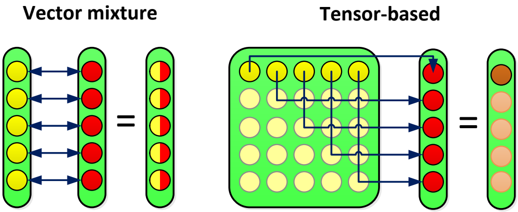
3 Tensor-based Models

4 Neural Models

5 Afterword

Relational words as functions

- In a vector mixture model, an adjective is of the same order as the noun it modifies, and both contribute equally to the result.
- One step further: Relational words are multi-linear maps (tensors of various orders) that can be applied to one or more arguments (vectors).



- Formalized in the context of compact closed categories by Coecke, Sadrzadeh and Clark (2010).

Quantizing the grammar

Coecke, Sadrzadeh and Clark (2010):

Pregroup grammars are structurally homomorphic with the category of finite-dimensional vector spaces and linear maps (both share compact closure)

- In abstract terms, there exists a structure-preserving passage from grammar to meaning:

$$\mathcal{F} : \text{Grammar} \to \text{Meaning}$$

- The meaning of a sentence $w_1 w_2 \dots w_n$ with grammatical derivation α is defined as:

$$\overrightarrow{w_1 w_2 \dots w_n} := \mathcal{F}(\alpha)(\overrightarrow{w_1} \otimes \overrightarrow{w_2} \otimes \dots \otimes \overrightarrow{w_n})$$

Pregroup grammars

A pregroup grammar $P(\Sigma, \mathcal{B})$ is a relation that assigns grammatical types from a pregroup algebra freely generated over a set of atomic types $\mathcal{B}$ to words of a vocabulary Σ.

- A pregroup algebra is a partially ordered monoid, where each element p has a left and a right adjoint such that:

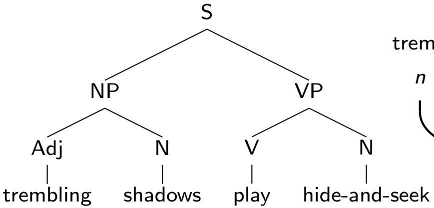
$$p \cdot p^r \leq 1 \leq p^r \cdot p \qquad p^l \cdot p \leq 1 \leq p \cdot p^l$$

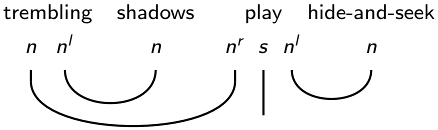
- Elements of the pregroup are basic (atomic) grammatical types, e.g. $\mathcal{B} = \{n, s\}$.
- Atomic grammatical types can be combined to form types of higher order (e.g. $n \cdot n^l$ or $n^r \cdot s \cdot n^l$)
- A sentence $w_1 w_2 \ldots w_n$ (with word w_i to be of type t_i) is grammatical whenever:

$$t_1 \cdot t_2 \cdot \ldots \cdot t_n \leq s$$

Pregroup derivation: example

$$p \cdot p^r \leq 1 \leq p^r \cdot p \qquad p^l \cdot p \leq 1 \leq p \cdot p^l$$





$$\begin{aligned} n \cdot n^l \cdot n \cdot n^r \cdot s \cdot n^l \cdot n \quad &\leq \quad n \cdot 1 \cdot n^r \cdot s \cdot 1 \\ &= \quad n \cdot n^r \cdot s \\ &\leq \quad 1 \cdot s \\ &= \quad s \end{aligned}$$

Compact closed categories

- A monoidal category $(\mathcal{C}, \otimes, I)$ is compact closed when every object has a left and a right adjoint, for which the following morphisms exist:

$$A \otimes A^r \xrightarrow{\epsilon^r} I \xrightarrow{\eta^r} A^r \otimes A \qquad\qquad A^l \otimes A \xrightarrow{\epsilon^l} I \xrightarrow{\eta^l} A \otimes A^l$$

- Pregroup grammars are CCCs, with ϵ and η maps corresponding to the partial orders
- **FdVect**, the category of finite-dimensional vector spaces and linear maps, is a also a (symmetric) CCC:
  - ϵ maps correspond to inner product
  - η maps to identity maps and multiples of those

A functor from syntax to semantics

We define a strongly monoidal functor $\mathcal{F}$ such that:

$$\mathcal{F} : P(\Sigma, \mathcal{B}) \to \textbf{FdVect}$$

$$\begin{aligned}
\mathcal{F}(p) &= P \quad \forall p \in \mathcal{B} \\
\mathcal{F}(1) &= \mathbb{R} \\
\mathcal{F}(p \cdot q) &= \mathcal{F}(p) \otimes \mathcal{F}(q) \\
\mathcal{F}(p^r) = \mathcal{F}(p^l) &= \mathcal{F}(p) \\
\mathcal{F}(p \leq q) &= \mathcal{F}(p) \to \mathcal{F}(q) \\
\mathcal{F}(\epsilon^r) = \mathcal{F}(\epsilon^l) &= \text{inner product in } \textbf{FdVect} \\
\mathcal{F}(\eta^r) = \mathcal{F}(\eta^l) &= \text{identity maps in } \textbf{FdVect}
\end{aligned}$$

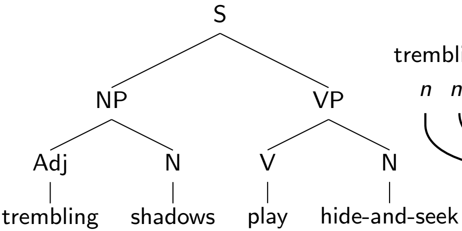
A multi-linear model

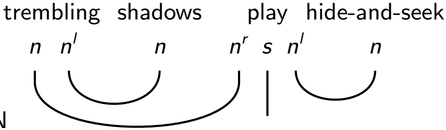
The grammatical type of a word defines the vector space in which the word lives:

- Nouns are vectors in N;
- adjectives are linear maps $N \rightarrow N$, i.e elements in $N \otimes N$;
- intransitive verbs are linear maps $N \rightarrow S$, i.e. elements in $N \otimes S$;
- transitive verbs are bi-linear maps $N \otimes N \rightarrow S$, i.e. elements of $N \otimes S \otimes N$;

- The composition operation is tensor contraction, i.e. elimination of matching dimensions by application of inner product.

Categorical composition: example





Type reduction morphism:

$$(\epsilon_n^r \cdot 1_s) \circ (1_n \cdot \epsilon_n^l \cdot 1_{n^r \cdot s} \cdot \epsilon_n^l) : n \cdot n^l \cdot n \cdot n^r \cdot s \cdot n^l \cdot n \to s$$

$$\mathcal{F}\left[(\epsilon_n^r \cdot 1_s) \circ (1_n \cdot \epsilon_n^l \cdot 1_{n^r \cdot s} \cdot \epsilon_n^l)\right] \left(\overline{trembling} \otimes \overrightarrow{shadows} \otimes \overline{play} \otimes \overrightarrow{hide\text{-}and\text{-}seek}\right) =$$

$$(\epsilon_N \otimes 1_S) \circ (1_N \otimes \epsilon_N \otimes 1_{N \otimes S} \otimes \epsilon_N) \left(\overline{trembling} \otimes \overrightarrow{shadows} \otimes \overline{play} \otimes \overrightarrow{hide\text{-}and\text{-}seek}\right) =$$

$$\overline{trembling} \times \overrightarrow{shadows} \times \overline{play} \times \overrightarrow{hide\text{-}and\text{-}seek}$$

$$\overrightarrow{shadows}, \overrightarrow{hide\text{-}and\text{-}seek} \in N \qquad \overline{trembling} \in N \otimes N \qquad \overline{play} \in N \otimes S \otimes N$$

Creating relational tensors: Extensional approach

A relational word is defined as the set of its arguments:

$$[\![red]\!] = \{car, door, dress, ink, \cdots\}$$

- Grefenstette and Sadrzadeh (2011):

$$\overline{adj} = \sum_i \overrightarrow{noun}_i \qquad \overline{verb}_{int} = \sum_i \overrightarrow{subj}_i \qquad \overline{verb}_{tr} = \sum_i \overrightarrow{subj}_i \otimes \overrightarrow{obj}_i$$

- Kartsaklis and Sadrzadeh (2016):

$$\overline{adj} = \sum_i \overrightarrow{noun}_i \otimes \overrightarrow{noun}_i \qquad \overline{verb}_{int} = \sum_i \overrightarrow{subj}_i \otimes \overrightarrow{subj}_i$$

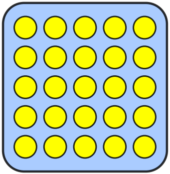
$$\overline{verb}_{tr} = \sum_i \overrightarrow{subj}_i \otimes \left(\frac{\overrightarrow{subj}_i + \overrightarrow{obj}_i}{2} \right) \otimes \overrightarrow{obj}_i$$

Creating relational tensors: Statistical approach

Baroni and Zamparelli (2010):

Create holistic distributional vectors for whole compounds (as if they were words) and use them to train a linear regression model.

red



$\times\ \overrightarrow{\text{car}} = \overrightarrow{\text{red car}}$

$\times\ \overrightarrow{\text{door}} = \overrightarrow{\text{red door}}$

$\times\ \overrightarrow{\text{dress}} = \overrightarrow{\text{red dress}}$

$\times\ \overrightarrow{\text{ink}} = \overrightarrow{\text{red ink}}$

$$\hat{\overline{adj}} = \arg\min_{\overline{adj}} \left[\frac{1}{2m} \sum_i (\overline{adj} \times \overrightarrow{noun_i} - \overrightarrow{adj\ noun_i})^2 \right]$$

Functional words

- Certain classes of words, such as determiners, relative pronouns, prepositions, or coordinators occur in almost every possible context.
- Thus, they are considered semantically vacuous from a distributional perspective and most often they are simply ignored.

In the tensor-based setting, these special words can be modelled by exploiting additional mathematical structures, such as Frobenius algebras and bialgebras.

Frobenius algebras in **FdVect**

- Given a symmetric CCC $(\mathcal{C}, \otimes, I)$, an object $X \in \mathcal{C}$ has a Frobenius structure on it if there exist morphisms:

  $\Delta : X \to X \otimes X \ , \ \iota : X \to I$ and $\mu : X \otimes X \to X \ , \ \zeta : I \to X$

  conforming to the Frobenius condition:

$$(\mu \otimes 1_X) \circ (1_X \otimes \Delta) = \Delta \circ \mu = (1_X \otimes \mu) \circ (\Delta \otimes 1_X)$$

- In **FdVect**, any vector space V with a fixed basis $\{\overrightarrow{v_i}\}_i$ has a commutative special Frobenius algebra over it [Coecke and Pavlovic, 2006]:

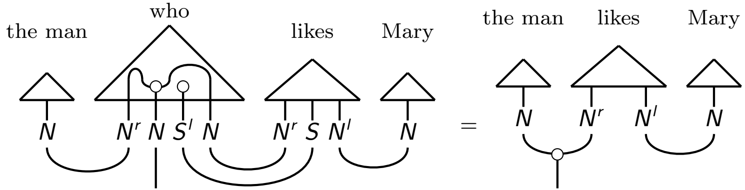
$$\Delta : \overrightarrow{v_i} \mapsto \overrightarrow{v_i} \otimes \overrightarrow{v_i} \qquad \mu : \overrightarrow{v_i} \otimes \overrightarrow{v_i} \mapsto \overrightarrow{v_i}$$

- It can be seen as copying and merging of the basis.

Frobenius algebras: Relative pronouns

How to represent relative pronouns in a tensor-based setting?

- A relative clause modifies the head noun of a phrase:



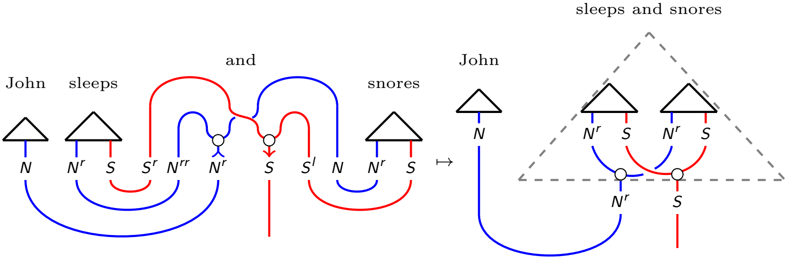
- The result is a merging of the vectors of the noun and the relative clause:

$$\overrightarrow{man} \odot (\overline{likes} \times \overrightarrow{Mary})$$

[Sadrzadeh, Clark, Coecke (2013)]

Frobenius algebras: Coordination

Copying and merging are the key processes in coordination:



- The subject is copied by a Δ-map and interacts individually with the two verbs
- The results are merged together with a μ-map

$$\overrightarrow{John}^{\mathsf{T}} \times (\overline{sleep} \odot \overline{snore})$$

[Kartsaklis (2016)]

Tensor-based models: Intuition

Tensor-based composition goes beyond a simple compatibility check between the two argument vectors; it *transforms* the input into an output of possibly different type.

- A verb, for example, is a function that takes as input a noun and transforms it into a sentence:

$$f_{int} : N \to S \qquad f_{tr} : N \times N \to S$$

- Size and form of the sentence space become tunable parameters of the models, and can depend on the task.
- Taking $S = \{\binom{0}{1}, \binom{1}{0}\}$, for example, provides an equivalent to formal semantics.

Tensor-based models: Pros and Cons

Distinguishing feature:

Relational words are multi-linear maps acting on arguments

PROS:

- Aligned with the formal semantics perspective
- More powerful than vector mixtures
- Flexible regarding the representation of functional words, such as relative pronouns and prepositions.

CONS:

- *Every* logical and functional word must be assigned to an appropriate tensor representation–it's not always clear how
- Space complexity problems for functions of higher arity (e.g. a ditransitive verb is a tensor of order 4)

Outline

1 Introduction

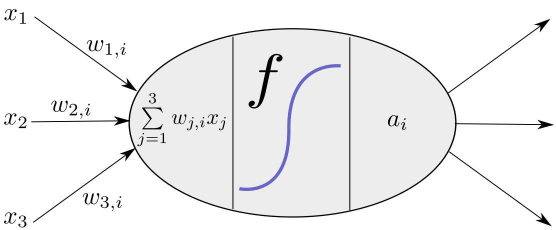
2 Vector Mixture Models

3 Tensor-based Models

4 Neural Models

5 Afterword

An artificial neuron

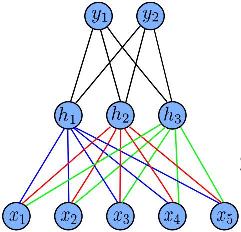


- The x_is form the input vector
- The w_{ji}s is a set of weights associated with the i-th output of the layer
- f is a non-linear function such as tanh or sigmoid
- a_i is the i-th output of the layer, computed as:

$$a_i = f(w_{1i}x_1 + w_{2i}x_2 + w_{3i}x_3)$$

A simple neural net

- A feed-forward neural network with one hidden layer:



$$h_1 = f(w_{11}x_1 + w_{21}x_2 + w_{31}x_3 + w_{41}x_4 + w_{51}x_5 + b_1)$$
$$h_2 = f(w_{12}x_1 + w_{22}x_2 + w_{32}x_3 + w_{42}x_4 + w_{52}x_5 + b_2)$$
$$h_3 = f(w_{13}x_1 + w_{23}x_2 + w_{33}x_3 + w_{43}x_4 + w_{53}x_5 + b_3)$$

or $\overrightarrow{h} = f(\mathbf{W}^{(1)}\overrightarrow{x} + \overrightarrow{b}^{(1)})$

Similarly:

$$\overrightarrow{y} = f(\mathbf{W}^{(2)}\overrightarrow{h} + \overrightarrow{b}^{(2)})$$

- Note that $\mathbf{W}^{(1)} \in \mathbb{R}^{3\times 5}$ and $\mathbf{W}^{(2)} \in \mathbb{R}^{2\times 3}$
- f is a non-linear function such as tanh or sigmoid (take $f = \mathrm{Id}$ and you have a tensor-based model)
- A universal approximator

Objective functions

- The goal of NN training is to find the set of parameters that optimizes a given objective function
- Or, to put it differently, to minimize an error function.
- Assume, for example, the goal of the NN is to produce a vector $\overrightarrow{y}$ that matches a specific target vector $\overrightarrow{t}$. The function:

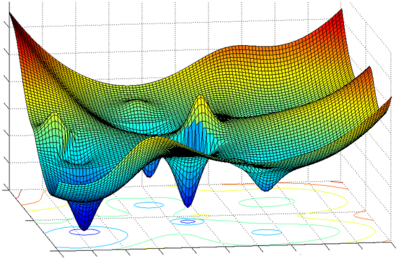
$$E = \frac{1}{2m} \sum_{i} || \overrightarrow{t_i} - \overrightarrow{y_i} ||^2$$

gives the total error across all training instances.

- We want to set the weights of the NN such that E becomes zero or as close to zero as possible.

Gradient descent

Take steps proportional to the *negative* of the gradient of E at the current point.



$$\Theta_t = \Theta_{t-1} - \alpha \nabla E(\Theta_{t-1})$$

- Θ_t: the parameters of the model at time step t
- α: a learning rate

(Graph taken from "The Beginner Programmer" blog, http://firsttimeprogrammer.blogspot.co.uk)

Backpropagation of errors

- How do we compute the error terms at the inner layers?

These are computed based one the errors of the next layer by using backpropagation. In general:

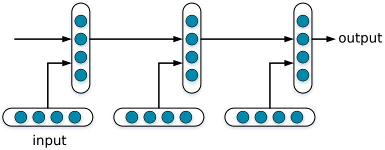
$$\delta_k = \Theta_k^{\mathsf{T}} \delta_{k+1} \odot f'(z_k)$$

- δ_k is the error vector at layer k
- Θ_k is the weight matrix of layer k
- z_k is the weighted sum at the output of layer k
- f' is the derivative of the non-linear function f

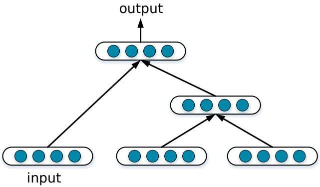
- Just an application of the chain rule for derivatives.

Recurrent and recursive NNs

- Standard NNs assume that inputs are independent of each other
- That is not the case in language; a word, for example, always depends on the previous words in the same sentence
- In a recurrent NN, connections form a directed cycle so that each output depends on the previous ones
- A recursive NN is applied recursively following a specific structure.



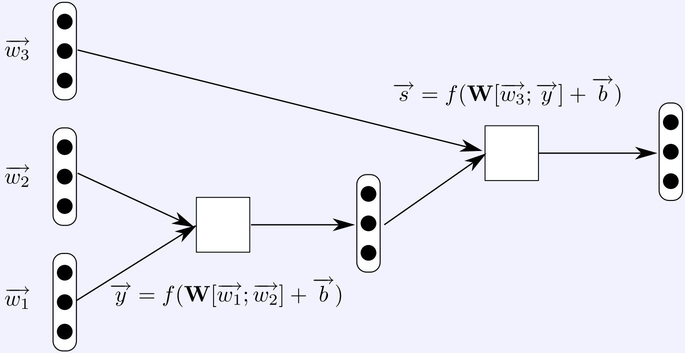
Recurrent NN



Recursive NN

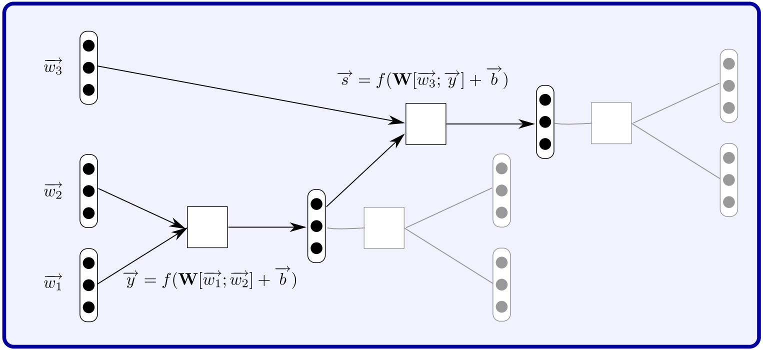
Recursive neural networks for composition

Pollack (1990); Socher et al. (2011;2012):



Unsupervised learning with NNs

- How can we train a NN in an unsupervised manner?
- Train the network to reproduce its input via an expansion layer:



- Use the output of the hidden layer as a compressed version of the input [Socher et al. (2011)]

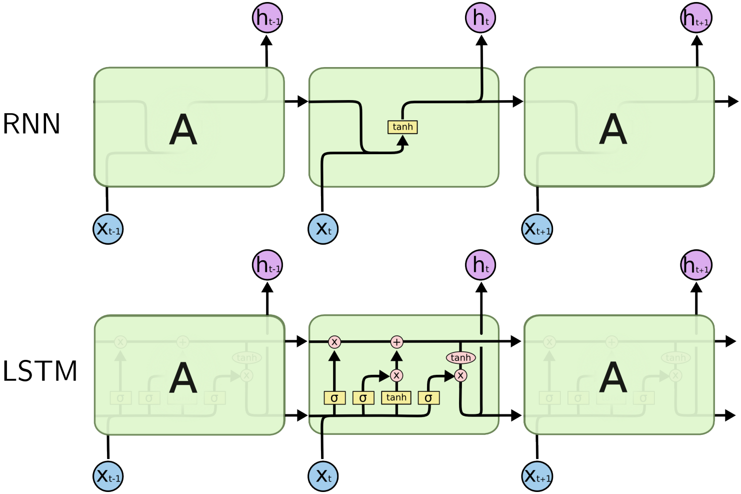
Long Short-Term Memory networks (1/2)

- RNNs are effective, but fail to capture long-range dependencies such as:

> **The movie I** liked and John said Mary and Ann really **hated**

- "Vanishing gradient" problem: Back-propagating the error requires the multiplication of many very small numbers together, and training for the bottom layers starts to stall.
- Long Short-Term Memory networks (LSTMs) (Hochreiter and Schmidhuber, 1997) provide a solution, by equipping each neuron with an internal state.

Long Short-Term Memory networks (2/2)



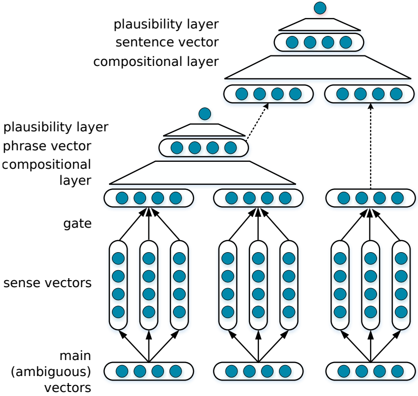
(Diagrams taken from Christopher Olah's blog, http://colah.github.io/)

Linguistically aware NNs

NN-based methods come mainly from image processing. How can we make them more linguistically aware?

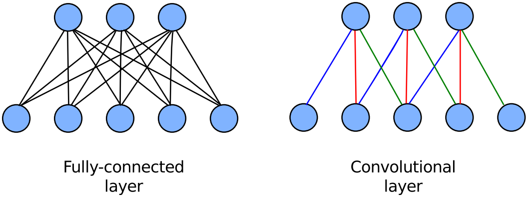
Cheng and Kartsaklis (2015):

- Take into account syntax, by optimizing against a scrambled version of each sentence
- Dynamically disambiguate the meaning of words during training based on their context



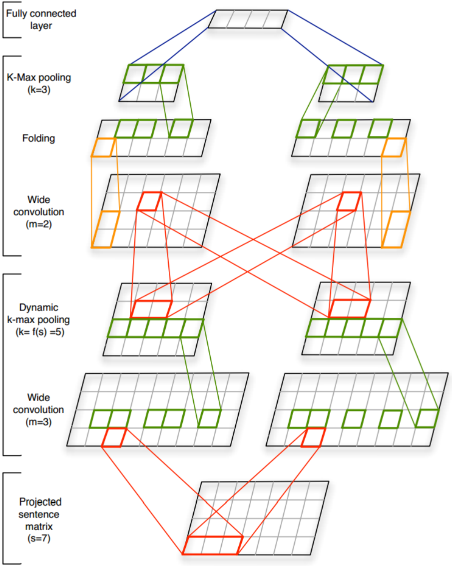
Convolutional NNs

- Originated in pattern recognition [Fukushima, 1980]
- Small filters apply on every position of the input vector:



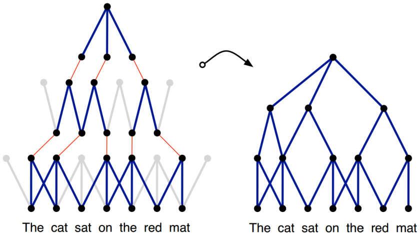
- Capable of extracting fine-grained local features independently of the exact position in input
- Features become increasingly global as more layers are stacked
- Each convolutional layer is usually followed by a pooling layer
- Top layer is fully connected, usually a soft-max classifier
- Application to language: Collobert and Weston (2008)

DCNNs for modelling sentences



Kalchbrenner, Grefenstette and Blunsom (2014): A deep architecture using dynamic k-max pooling

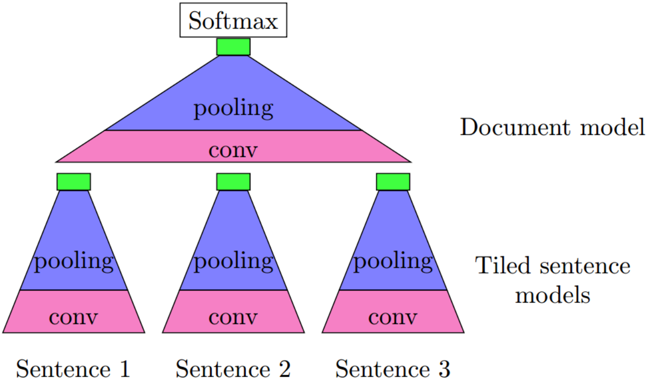
- Syntactic structure is induced automatically:



(*Figures reused with permission*)

Beyond sentence level

An additional convolutional layer can provide document vectors [Denil et al. (2014)]:



(*Figure reused with permission*)

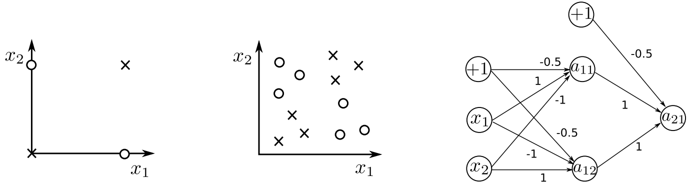
Neural models: Intuition (1/2)

- Recall that tensor-based composition involves a linear transformation of the input into some output.
- Neural models make this process more effective by applying consecutive non-linear layers of transformation.

A NN does not only project a noun vector onto a sentence space, but it can also transform the geometry of the space itself in order to make it reflect better the relationships between the points (sentences) in it.

Neural models: Intuition (2/2)

- **Example:** Although there is no linear map to send an input $x \in \{0, 1\}$ to the correct XOR value, the function can be approximated by a simple NN with one hidden layer.



(a) (b) (c)

- Points in (b) can be seen as representing two semantically distinct groups of sentences, which the NN is able to distinguish (while a linear map cannot)

Neural models: Pros and Cons

Distinguishing feature:

Drastic transformation of the sentence space.

PROS:

- Non-linearity and layered approach allow the simulation of a very wide range of functions
- Word vectors are parameters of the model, optimized during training
- State-of-the-art results in a number of NLP tasks

CONS:

- Requires expensive training based on backpropagation
- Difficult to discover the right configuration
- A "black-box" approach: not easy to correlate inner workings with output

Outline

1. Introduction
2. Vector Mixture Models
3. Tensor-based Models
4. Neural Models
5. Afterword

Refresher: A CDMs hierarchy

		Distinguishing feature	Linguistic justification	Space complexity	Training time
Non-linear	Deep learning models	Drastic transformation of sentence space geometry	Med	Med	High
Multi-linear	Tensor-based models	Relational words are functions applied on argument vectors	High	High	Med
Element-wise mixing	Vector mixture models	Equal contribution of all words	Low	Low	—

Theoretical power

Open issues-Future work

- No convincing solution for logical connectives, negation, quantifiers and so on.
- Functional words, such as prepositions and relative pronouns, are also a problem.
- Sentence space is usually identified with word space. This is convenient, but is it the right thing to do?
- Solutions depend on the specific CDM class—e.g. not much to do in a vector mixture setting
- **Important:** How can we make NNs more linguistically aware? [Cheng and Kartsaklis (2015)]

Summary

- CDMs provide quantitative semantic representations for sentences (or even documents)
- Element-wise operations on word vectors constitute an easy and reasonably effective way to get sentence vectors
- Categorical compositional distributional models allow reasoning on a theoretical level—a glass box approach
- Neural models are extremely powerful and effective; still a black-box approach, not easy to explain why a specific configuration works and some other does not.
- Convolutional networks seem to constitute the most promising solution to the problem of capturing the meaning of sentences

Thank you for your attention!

References I

Baroni, M. and Zamparelli, R. (2010).
Nouns are Vectors, Adjectives are Matrices.
In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Cheng, J. and Kartsaklis, D. (2015).
Syntax-aware multi-sense word embeddings for deep compositional models of meaning.
In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1531–1542, Lisbon, Portugal. Association for Computational Linguistics.

Coecke, B., Sadrzadeh, M., and Clark, S. (2010).
Mathematical Foundations for a Compositional Distributional Model of Meaning. Lambek Festschrift.
Linguistic Analysis, 36:345–384.

Collobert, R. and Weston, J. (2008).
A unified architecture for natural language processing: Deep neural networks with multitask learning.
In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Denil, M., Demiraj, A., Kalchbrenner, N., Blunsom, P., and de Freitas, N. (2014).
Modelling, visualising and summarising documents with a single convolutional neural network.
Technical Report arXiv:1406.3830, University of Oxford.

Grefenstette, E. and Sadrzadeh, M. (2011).
Experimental support for a categorical compositional distributional model of meaning.
In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1394–1404, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Harris, Z. (1968).
Mathematical Structures of Language.
Wiley.

References II

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014).
A convolutional neural network for modelling sentences.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland. Association for Computational Linguistics.

Kartsaklis, D. (2015).
Compositional Distributional Semantics with Compact Closed Categories and Frobenius Algebras.
PhD thesis, University of Oxford.

Kartsaklis, D. and Sadrzadeh, M. (2016).
A compositional distributional inclusion hypothesis.
In *Proceedings of the 2017 Conference on Logical Aspects of Computational Linguistics*, Nancy, France. Springer.

Mitchell, J. and Lapata, M. (2010).
Composition in distributional models of semantics.
Cognitive Science, 34(8):1388–1439.

Montague, R. (1970a).
English as a formal language.
In *Linguaggi nella Società e nella Tecnica*, pages 189–224. Edizioni di Comunità, Milan.

Montague, R. (1970b).
Universal grammar.
Theoria, 36:373–398.

Sadrzadeh, M., Clark, S., and Coecke, B. (2013).
The Frobenius anatomy of word meanings I: subject and object relative pronouns.
Journal of Logic and Computation, Advance Access.

References III

Sadrzadeh, M., Clark, S., and Coecke, B. (2014).
The Frobenius anatomy of word meanings II: Possessive relative pronouns.
Journal of Logic and Computation.

Socher, R., Huang, E., Pennington, J., Ng, A., and Manning, C. (2011).
Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection.
Advances in Neural Information Processing Systems, 24.

Socher, R., Huval, B., Manning, C., and A., N. (2012).
Semantic compositionality through recursive matrix-vector spaces.
In *Conference on Empirical Methods in Natural Language Processing 2012.*

Socher, R., Manning, C., and Ng, A. (2010).
Learning Continuous Phrase Representations and Syntactic Parsing with recursive neural networks.
In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop.*