

Deep Learning for Semantic Composition

Xiaodan Zhu* & Edward Grefenstette[†]

*National Research Council Canada
Queen's University
zhu2048@gmail.com

[†]DeepMind
etg@google.com

July 30th, 2017

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

Principle of Compositionality

Principle of compositionality: The meaning of a whole is a function of the meaning of the parts.

Principle of Compositionality

Principle of compositionality: The meaning of a whole is a function of the meaning of the parts.

- While we focus on natural language, compositionality exists not just in language.

Principle of compositionality: The meaning of a whole is a function of the meaning of the parts.

- While we focus on natural language, compositionality exists not just in language.
 - Sound/music
 - Music notes are composed with some regularity but not randomly arranged to form a song.

Principle of compositionality: The meaning of a whole is a function of the meaning of the parts.

- While we focus on natural language, compositionality exists not just in language.
 - Sound/music
 - Music notes are composed with some regularity but not randomly arranged to form a song.
 - Vision
 - Natural scenes are composed of meaningful components.
 - Artificial visual art pieces often convey certain meaning with regularity from their parts.

Principle of Compositionality

Compositionality is regarded by many as a fundamental component of intelligence in addition to language understanding (Miller et al., 1976; Fodor et al., 1988; Bienenstock et al., 1996; Lake et al., 2016).

Compositionality is regarded by many as a fundamental component of intelligence in addition to language understanding (Miller et al., 1976; Fodor et al., 1988; Bienenstock et al., 1996; Lake et al., 2016).

- For example, Lake et al. (2016) emphasize several essential ingredients for building machines that “*learn and think like people*”:
 - Compositionality
 - Intuitive physics/psychology
 - Learning-to-learn
 - Causality models

Compositionality is regarded by many as a fundamental component of intelligence in addition to language understanding (Miller et al., 1976; Fodor et al., 1988; Bienenstock et al., 1996; Lake et al., 2016).

- For example, Lake et al. (2016) emphasize several essential ingredients for building machines that “*learn and think like people*”:
 - Compositionality
 - Intuitive physics/psychology
 - Learning-to-learn
 - Causality models
- Note that many of these challenges present in natural language understanding.

Compositionality is regarded by many as a fundamental component of intelligence in addition to language understanding (Miller et al., 1976; Fodor et al., 1988; Bienenstock et al., 1996; Lake et al., 2016).

- For example, Lake et al. (2016) emphasize several essential ingredients for building machines that “*learn and think like people*”:
 - Compositionality
 - Intuitive physics/psychology
 - Learning-to-learn
 - Causality models
- Note that many of these challenges present in natural language understanding.
 - They are reflected in the sparseness in training a NLP model.

Compositionality is regarded by many as a fundamental component of intelligence in addition to language understanding (Miller et al., 1976; Fodor et al., 1988; Bienenstock et al., 1996; Lake et al., 2016).

- For example, Lake et al. (2016) emphasize several essential ingredients for building machines that “*learn and think like people*”:
 - Compositionality
 - Intuitive physics/psychology
 - Learning-to-learn
 - Causality models
- Note that many of these challenges present in natural language understanding.
 - They are reflected in the sparseness in training a NLP model.
- Note also that compositionality may be entangled with the other “ingredients” listed above.

Semantic Composition in Natural Language

- *good* \rightarrow *very good* \rightarrow *not very good* \rightarrow ...

Semantic Composition in Natural Language

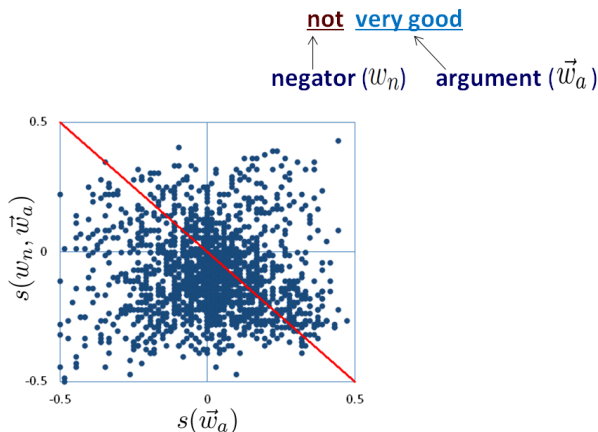
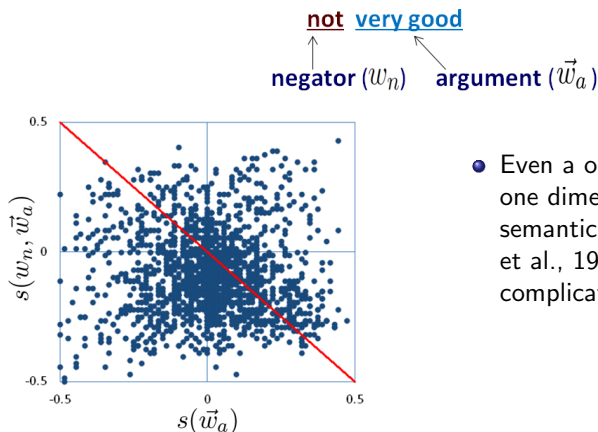


Figure: Results from (Zhu et al., 2014).
A dot in the figure corresponds to a negated phrase (e.g., not very good) in Stanford Sentiment Treebank (Socher et al., 2013).
The y-axis is its sentiment value and x-axis the sentiment of its argument.

Semantic Composition in Natural Language



- Even a one-layer composition, over one dimension of meaning (e.g., semantic orientation (Osgood et al., 1957)), could be a complicated mapping.

Figure: Results from (Zhu et al., 2014).
A dot in the figure corresponds to a negated phrase (e.g., not very good) in Stanford Sentiment Treebank (Socher et al., 2013). The y-axis is its sentiment value and x-axis the sentiment of its argument.

Semantic Composition in Natural Language

- *good* → *very good* → *not very good* → ...
- *senator* → *former senator* → ...
- *basketball player* → *short basketball player* → ...
- *giant* → *small giant* → ...
- *empty/full* → *half empty/full* → *almost half empty/full* → ...¹

¹See more examples in (Partee, 1995).

Semantic Composition in Natural Language

Semantic composition in natural language: the task of **modelling** the meaning of **a larger piece of text** by composing the meaning of its **constituents**.

Semantic composition in natural language: the task of **modelling** the meaning of **a larger piece of text** by composing the meaning of its **constituents**.

- *modelling*: learning a representation
 - The compositionality in language is very challenging as discussed above.
 - Compositionality can entangle with other challenges such as those emphasized in (Lake et al., 2016).

Semantic composition in natural language: the task of **modelling** the meaning of **a larger piece of text** by composing the meaning of its **constituents**.

- *modelling*: learning a representation
 - The compositionality in language is very challenging as discussed above.
 - Compositionality can entangle with other challenges such as those emphasized in (Lake et al., 2016).
- *a larger piece of text*: a phrase, sentence, or document.

Semantic composition in natural language: the task of **modelling** the meaning of **a larger piece of text** by composing the meaning of its **constituents**.

- *modelling*: learning a representation
 - The compositionality in language is very challenging as discussed above.
 - Compositionality can entangle with other challenges such as those emphasized in (Lake et al., 2016).
- *a larger piece of text*: a phrase, sentence, or document.
- *constituents*: subword components, words, phrases.

Semantic composition in natural language: the task of **modelling** the meaning of **a larger piece of text** by composing the meaning of its **constituents**.

- *modelling*: learning a representation
 - The compositionality in language is very challenging as discussed above.
 - Compositionality can entangle with other challenges such as those emphasized in (Lake et al., 2016).
- *a larger piece of text*: a phrase, sentence, or document.
- *constituents*: subword components, words, phrases.

Introduction

Two key problems:

- How to represent meaning?
- How to learn such a representation?

Representation

Let's first very briefly revisit the **representation** we assume in this tutorial ... and leave the **learning** problem to the entire tutorial that follows.

Representation

Let's first very briefly revisit the **representation** we assume in this tutorial ... and leave the **learning** problem to the entire tutorial that follows.

Love

Love:

a (1) : strong affection for another arising out of kinship or personal ties
<maternal *love* for a child> (2) : attraction based on sexual desire : affection
and tenderness felt by **lovers** (3) : affection based on admiration, benevolence,
or common interests <*love* for his old schoolmates>

... ..

— *The Merriam-Webster Dictionary*

Love:

a (1) : strong affection for another arising out of kinship or personal ties
<maternal /love for a child> (2) : attraction based on sexual desire : affection
and tenderness felt by **lovers** (3) : affection based on admiration, benevolence,
or common interests </love for his old schoolmates>

... ..

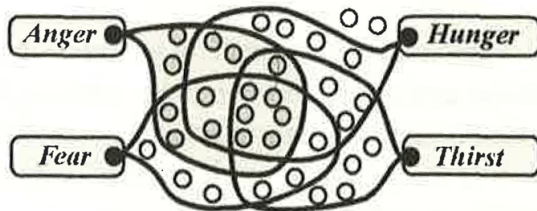
— *The Merriam-Webster Dictionary*

love, admiration, satisfaction ...

anger, fear, hunger ...

Representation

A viewpoint from *The Emotion Machine* (Minsky, 2006)



Representation

A viewpoint from *The Emotion Machine* (Minsky, 2006)



- Each variable responds to different concepts and each concept is represented by different variables.

Representation

A viewpoint from *The Emotion Machine* (Minsky, 2006)



- Each variable responds to different concepts and each concept is represented by different variables.
 - This is exactly a *distributed representation*.

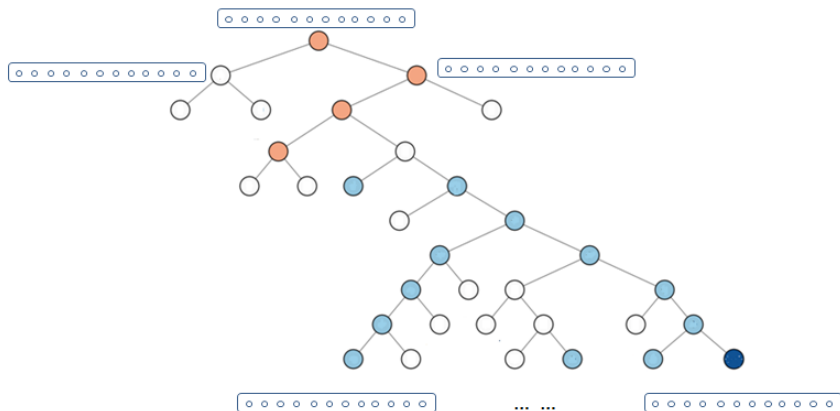
love



“You should know a word by the
company it keeps” (Firth, 1957)

Modelling Composition Functions

How do we model the composition functions?



Deep Learning for Semantic Composition

Deep learning: We focus on deep learning models in this tutorial.

Deep Learning for Semantic Composition

Deep learning: We focus on deep learning models in this tutorial.

“Wait a minute, deep learning again?”

“DL people, leave language along ...”

Deep Learning for Semantic Composition

Deep learning: We focus on deep learning models in this tutorial.

“Wait a minute, deep learning again?”

“DL people, leave language along ...”

Asking some questions may be helpful:

- Are deep learning models providing nice function or density approximation, the problems that many specific NLP tasks essentially seek to solve? $X \rightarrow Y$
- Are continuous vector representations of meaning effective for (as least some) NLP tasks? Are DL models convenient for computing such continuous representations?
- Do DL models naturally bridge language with other modalities in terms of both representation and learning? (this could be important.)

More questions:

- What NLP problems (e.g., semantic problems here) can be better handled with DL and what cannot?
- Can NLP benefit from combining DL and other approaches (e.g., symbolic approaches)?
- In general, has the effectiveness of DL models for semantics already been well understood?

Deep Learning for Semantic Composition

1 Introduction

- Semantic composition
- **Formal methods**
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

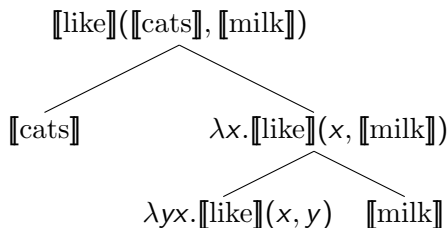
Montague Semantics (1970–1973):

- Treat natural language like a formal language via
 - an interpretation function $\llbracket \dots \rrbracket$, and
 - a mapping from CFG rules to function application order.
- Interpretation of a sentence reduces to logical form via β -reduction.

High Level Idea

Syntax guides composition, types determine their semantics, predicate logic does the rest.

Syntactic Analysis	Semantic Interpretation
$S \Rightarrow NP VP$	$\llbracket VP \rrbracket(\llbracket NP \rrbracket)$
$NP \Rightarrow \text{cats, milk, etc.}$	$\llbracket \text{cats} \rrbracket, \llbracket \text{milk} \rrbracket, \dots$
$VP \Rightarrow Vt NP$	$\llbracket Vt \rrbracket(\llbracket NP \rrbracket)$
$Vt \Rightarrow \text{like, hug, etc.}$	$\lambda yx. \llbracket \text{like} \rrbracket(x, y), \dots$



Cats like milk.

Pros:

- Intuitive and interpretable(?) representations.
- Leverage the power of predicate logic to model semantics.
- Evaluate the truth of statements, derive conclusions, etc.

Cons:

- Brittle, requires robust parsers.
- Extensive logical model required for evaluation of clauses.
- Extensive set of rules required to do anything useful.
- Overall, an intractable (or unappealing) learning problem.

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

Simple Parametric Models

Basic models with pre-defined function form (Mitchell et al., 2008):

General form : $\mathbf{p} = f(\mathbf{u}, \mathbf{v}, R, K)$

Add : $\mathbf{p} = \mathbf{u} + \mathbf{v}$

WeightAdd : $\mathbf{p} = \alpha^T \mathbf{u} + \beta^T \mathbf{v}$

Multiplicative : $\mathbf{p} = \mathbf{u} \otimes \mathbf{v}$

Combined : $\mathbf{p} = \alpha^T \mathbf{u} + \beta^T \mathbf{v} + \gamma^T (\mathbf{u} \otimes \mathbf{v})$

We will see later in this tutorial that the above models could be seen as special cases of more complicated composition models.

Results

Reference (R):	The <u>color</u> <u>ran</u> .
High-similarity landmark (H):	The <u>color</u> <u>dissolved</u> .
Low-similarity landmark (L):	The <u>color</u> <u>galloped</u> .

A good composition model should give the above R-H pair a similarity score higher than that given to the R-L pair. Also, a good model should assign such similarity scores with a high correlation (ρ) to what human assigned.

Results

Reference (R): The color ran.
High-similarity landmark (H): The color dissolved.
Low-similarity landmark (L): The color galloped.

A good composition model should give the above R-H pair a similarity score higher than that given to the R-L pair. Also, a good model should assign such similarity scores with a high correlation (ρ) to what human assigned.

Models	R-H similarity	R-L similarity	ρ
NonComp	0.27	0.26	0.08**
Add	0.59	0.59	0.04*
WeightAdd	0.35	0.34	0.09**
Kintsch	0.47	0.45	0.09**
Multiply	0.42	0.28	0.17**
Combined	0.38	0.28	0.19**
UpperBound	4.94	3.25	0.40**

Table: Mean cosine similarities for the R-H pairs and R-L pairs as well as the correlation coefficients (ρ) with human judgments (*: $p < 0.05$, **: $p < 0.01$).

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

Parameterizing Composition Functions

To move beyond simple algebraic or parametric models we need function approximators which, ideally:

- Can approximate any **arbitrary function** (e.g. ANNs).
- Can cope with **variable size** sequences.
- Can capture long range or unbounded **dependencies**.
- Can implicitly or explicitly model **structure**.
- Can be trained against a **supervised or unsupervised objective** (or both — semi-supervised training).
- Can be trained chiefly or primarily through **backpropagation**.

A Neural Network Model Zoo

This section presents a selection of models satisfying some (if not all) of these criteria.

Outline

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

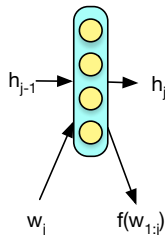
4 Summary

Recurrent Neural Networks

Bounded Methods

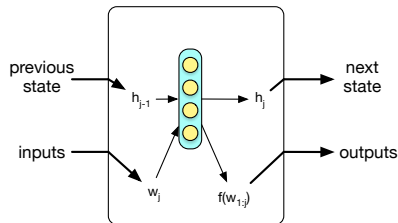
Many methods impose explicit or implicit length limits on conditioning information. For example:

- order- n Markov assumption in NLM/LBL
- fully-connected layers and dynamic pooling in conv-nets



Recurrent Neural Networks introduce a repeatedly composable unit, the *recurrent cell*, which both models an unbounded sequence prefix and express a function over it.

The Mathematics of Recurrence



Building Blocks

- An input vector $w_j \in \mathbb{R}^{|w|}$
- A previous state $h_{j-1} \in \mathbb{R}^{|h|}$
- A next state $h_j \in \mathbb{R}^{|h|}$
- An output $y_j \in \mathbb{R}^{|y|}$
- $f_y : \mathbb{R}^{|w|} \times \mathbb{R}^{|h|} \rightarrow \mathbb{R}^{|y|}$
- $f_h : \mathbb{R}^{|w|} \times \mathbb{R}^{|h|} \rightarrow \mathbb{R}^{|h|}$

Putting it together

$$h_j = f_h(w_j, h_{j-1})$$

$$y_j = f_y(w_j, h_j)$$

So $y_j = f_y(w_j, f_h(w_{j-1}, h_{j-1})) = f_y(w_j, f_h(w_{j-1}, f_h(w_{j-2}, h_{j-2}))) = \dots$

RNNs for Language Modelling

Language modelling

We want to model the joint probability of tokens t_1, \dots, t_n in a sequence:

$$P(t_1, \dots, t_n) = P(t_1) \prod_{i=2}^n P(t_i | t_1, \dots, t_{i-1})$$

Adapting a recurrence for basic LM

For vocab V , define an embedding matrix $E \in \mathbb{R}^{|V| \times |w|}$ and a logit projection matrix $W_V \in \mathbb{R}^{|V| \times |V|}$. Then:

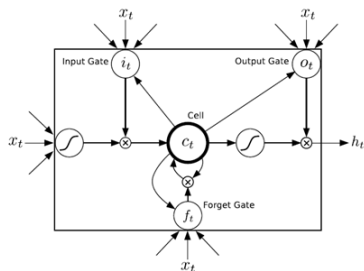
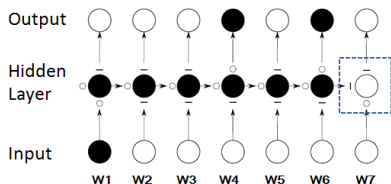
$$w_j = \text{embed}(t_j, E)$$

$$y_j = f_y(w_j, h_j) \quad h_j = f_h(w_j, h_{j-1})$$

$$p_j = \text{softmax}(y_j W_V)$$

$$P(t_{j+1} | t_1, \dots, t_j) = \mathbf{Categorical}(t_{j+1}; p_j)$$

Aside: The Vanishing Gradient Problem and LSTM RNNs

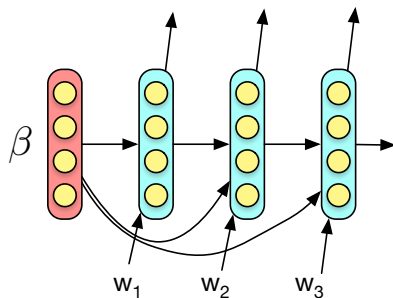
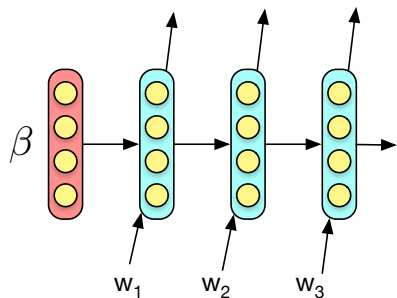


- RNN is deep “*by time*”, so it could seriously suffer from the vanishing gradient issue.
- LSTM configures memory cells and multiple “gates” to control information flow. If properly learned, LSTM can keep pretty long-distance (hundreds of time steps) information in memory.
- Memory-cell details:
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1})$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1})$$
$$c_t = \sigma(f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1}))$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t)$$
$$h_t = \sigma(o_t \tanh(c_t))$$

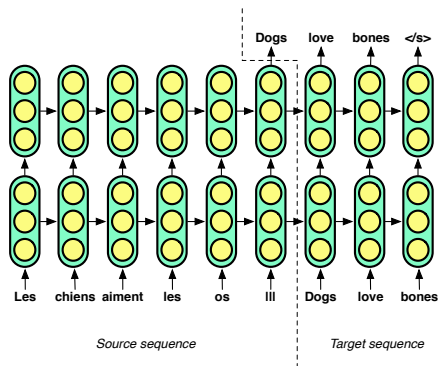
Conditional Language Models

Conditional Language Modelling

A strength of RNNs is that h_j can model not only the history of the generated/observed sequence t_1, \dots, t_j , but any conditioning information β , e.g. by setting $h_0 = \beta$.



Encoder-Decoder Models with RNNs



cf. Kalchbrenner et al., 2013; Sutskever et al., 2014

- Model $p(t_1, \dots, t_n | s_1, \dots, s_m)$
- $h_i^e = RNN_{encoder}(s_i, h_{i-1}^e)$
- $h_i^d = RNN_{decoder}(t_i, h_{i-1}^d)$
- $h_0^d = h_m^e$
- $t_{i+1} \sim \mathbf{Categorical}(t; f_V(h_i))$

The encoder RNN as a composition module

All information needed to transduce the source into the target sequence using $RNN_{decoder}$ needs to be present in the start state h_0^d .

This start state is produced by $RNN_{encoder}$, which will learn to compose.

RNNs as Sentence Encoders

This idea of RNNs as sentence encoder works for classification as well:

- Data is labelled sequences $(s_1, \dots, s_{|s|}; \hat{y})$.
- RNN is run over s to produce final state $h_{|s|} = RNN(s)$.
- A differentiable function of $h_{|s|}$ classifies: $y = f_{\theta}(h_{|s|})$
- $h_{|s|}$ can be taken to be the composed meaning of s , with regard to the task at hand.

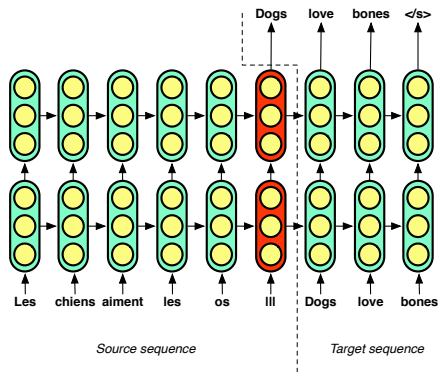
An aside: Bi-directional RNN encoders

For both sequence classification and generation, sometimes a Bi-directional RNN is used to encode:

$$h_i^{\leftarrow} = RNN^{\leftarrow}(s_i, h_{i+1}^{\leftarrow}) \quad h_i^{\rightarrow} = RNN^{\rightarrow}(s_i, h_{i-1}^{\rightarrow})$$

$$h_{|s|} = \text{concat}(h_1^{\leftarrow}, h_{|s|}^{\rightarrow})$$

A Transduction Bottleneck



Single vector representation of sentences causes problems:

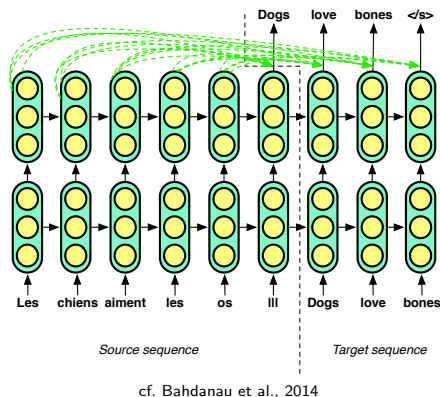
- Training focusses on learning marginal language model of target language first.
- Longer input sequences cause compressive loss.
- Encoder gets significantly diminished gradient.

In the words of Ray Mooney...

"You can't cram the meaning of a whole %&!\$ing sentence into a single \$&!*ing vector!"

Yes, the censored-out swearing is copied verbatim.

Attention



We want to use h_1^e, \dots, h_m^e when predicting t_i by conditioning on words that might *relate* to t_i :

- 1 Compute h_i^d (RNN update)
- 2 $e_{ij} = f_{att}(h_i^d, h_j^e)$
- 3 $a_{ij} = \text{softmax}(\mathbf{e}_i)_j$
- 4 $h_i^{att} = \sum_{j=1}^m a_{ij} h_j^e$
- 5 $\hat{h}_i = \text{concat}(h_i^d, h_i^{att})$
- 6 $t_{i+1} \sim \text{Categorical}(t; f_V(\hat{h}_i))$

The many faces of attention

Many variants on the above process: early attention (based on h_{i-1}^d and t_i , used to update h_i^d), different attentive functions f_{att} (e.g. based on projected inner products, or MLPs), and so on.

Attention and Composition

We refer to the set of source activation vectors h_1^e, \dots, h_m^e in the previous slides as an attention matrix. Is it a suitable sentence representation?

Pros:

- Locally compositional: vectors contain information about other words (especially with bi-directional RNN as encoder).
- Variable size sentence representation: longer sentences yield larger representation with more capacity.

Cons:

- Single vector representation of sentences is convenient (many decoders, classifiers, etc. expect fixed-width feature vectors as input)
- Locally compositional, but are long range dependencies resolved in the attention matrix? Does it truly express the sentence's meaning as a semantic unit (or is it just good for sequence transduction)?

Outline

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- **Recursive composition models**
- Convolutional composition models
- Unsupervised models

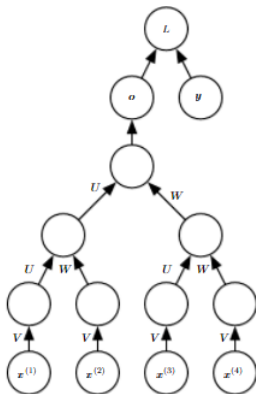
3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

Recursive Neural Networks

Recursive networks: a generalization of (chain) recurrent networks with a computational graph, often a tree (Pollack, 1990; Francesconi et al., 1997; Socher et al., 2011a,b,c, 2013; Zhu et al., 2015b)



Recursive Neural Networks

- Successfully applied to consider input data structures.
 - Natural language processing (Socher et al., 2011a,c; Le et al., 2015; Tai et al., 2015; Zhu et al., 2015b)
 - Computer vision (Socher et al., 2011b)

Recursive Neural Networks

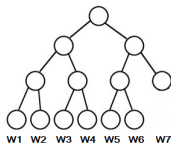
- Successfully applied to consider input data structures.
 - Natural language processing (Socher et al., 2011a,c; Le et al., 2015; Tai et al., 2015; Zhu et al., 2015b)
 - Computer vision (Socher et al., 2011b)
- How to determine the structures.
 - Encode given “external” knowledge about the structure of the input data,
 - e.g., syntactic structures; modelling sentential semantics and syntax is one of the most interesting problems in language.

Recursive Neural Networks

- Successfully applied to consider input data structures.
 - Natural language processing (Socher et al., 2011a,c; Le et al., 2015; Tai et al., 2015; Zhu et al., 2015b)
 - Computer vision (Socher et al., 2011b)
- How to determine the structures.
 - Encode given “external” knowledge about the structure of the input data,
 - e.g., syntactic structures; modelling sentential semantics and syntax is one of the most interesting problems in language.

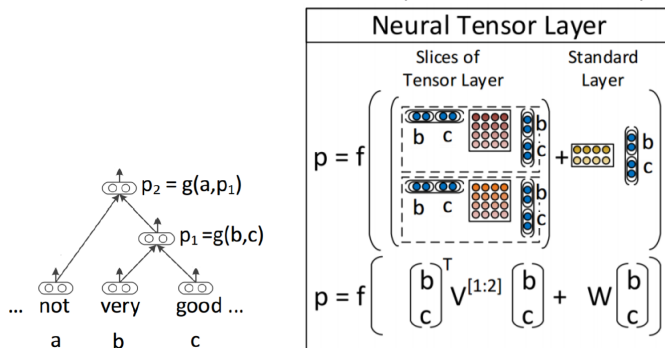
Recursive Neural Networks

- Successfully applied to consider input data structures.
 - Natural language processing (Socher et al., 2011a,c; Le et al., 2015; Tai et al., 2015; Zhu et al., 2015b)
 - Computer vision (Socher et al., 2011b)
- How to determine the structures.
 - Encode given “external” knowledge about the structure of the input data,
 - e.g., syntactic structures; modelling sentential semantics and syntax is one of the most interesting problems in language.
 - Encode simply a *complete* tree.



Integrating Syntactic Parses in Composition

Recursive Neural Tensor Network (Socher et al., 2012):



- The structure is given (here by a constituency parser.)
- Each node here is implemented as a regular feed-forward layer plus a 3^{rd} -order tensor.
 - The tensor captures 2^{nd} -degree (quadratic) polynomial interaction of children, e.g., b_i^2 , $b_i c_j$, and c_j^2 .

Results

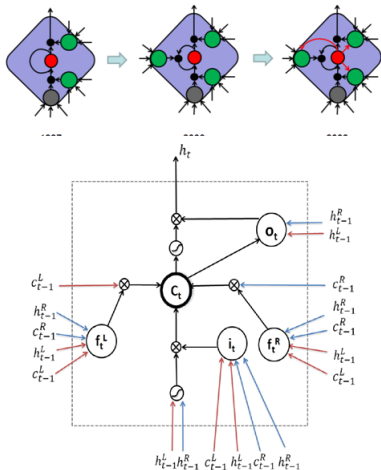
The models have been successfully applied to a number of tasks such as sentiment analysis (Socher et al., 2013).

Model	Fine-grained		Positive/Negative	
	All	Root	All	Root
NB	67.2	41.0	82.6	81.8
SVM	64.3	40.7	84.6	79.4
BiNB	71.0	41.9	82.7	83.1
VecAvg	73.3	32.7	85.1	80.1
RNN	79.0	43.2	86.1	82.4
MV-RNN	78.7	44.4	86.8	82.9
RNTN	80.7	45.7	87.6	85.4

Table: Accuracy for fine grained (5-class) and binary predictions at the sentence level (root) and for all nodes.

Tree-LSTM

Tree-structured LSTM (Le, *SEM-15; Tai, ACL-15; Zhu, ICML-15): It is an extension of chain LSTM to tree structures.



$$i_t = \sigma(W_{hi}^L h_{t-1}^L + W_{hi}^R h_{t-1}^R + W_{ci}^L c_{t-1}^L + W_{ci}^R c_{t-1}^R + b_i) \quad (1)$$

$$f_t^L = \sigma(W_{hf_i}^L h_{t-1}^L + W_{hf_i}^R h_{t-1}^R + W_{cf_i}^L c_{t-1}^L + W_{cf_i}^R c_{t-1}^R + b_{f_i}) \quad (2)$$

$$f_t^R = \sigma(W_{hf_r}^L h_{t-1}^L + W_{hf_r}^R h_{t-1}^R + W_{cf_r}^L c_{t-1}^L + W_{cf_r}^R c_{t-1}^R + b_{f_r}) \quad (3)$$

$$x_t = W_{hx}^L h_{t-1}^L + W_{hx}^R h_{t-1}^R + b_x \quad (4)$$

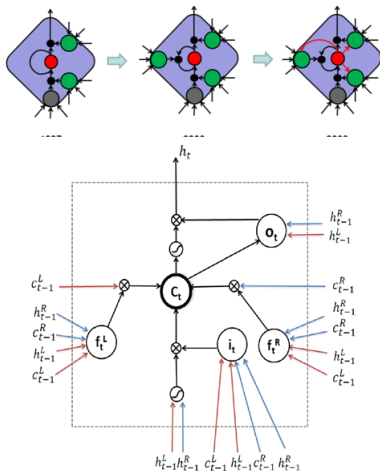
$$c_t = f_t^L c_{t-1}^L + f_t^R c_{t-1}^R + i_t \tanh(x_t) \quad (5)$$

$$o_t = \sigma(W_{ho}^L h_{t-1}^L + W_{ho}^R h_{t-1}^R + W_{co} c_t + b_o) \quad (6)$$

$$h_t = o_t \tanh(c_t) \quad (7)$$

Tree-LSTM

Tree-structured LSTM (Le, *SEM-15; Tai, ACL-15; Zhu, ICML-15): It is an extension of chain LSTM to tree structures.



$$i_t = \sigma(W_{hi}^L h_{t-1}^L + W_{hi}^R h_{t-1}^R + W_{ci}^L c_{t-1}^L + W_{ci}^R c_{t-1}^R + b_i) \quad (1)$$

$$f_t^L = \sigma(W_{hf_i}^L h_{t-1}^L + W_{hf_i}^R h_{t-1}^R + W_{cf_i}^L c_{t-1}^L + W_{cf_i}^R c_{t-1}^R + b_{f_i}) \quad (2)$$

$$f_t^R = \sigma(W_{hf_r}^L h_{t-1}^L + W_{hf_r}^R h_{t-1}^R + W_{cf_r}^L c_{t-1}^L + W_{cf_r}^R c_{t-1}^R + b_{f_r}) \quad (3)$$

$$x_t = W_{hx}^L h_{t-1}^L + W_{hx}^R h_{t-1}^R + b_x \quad (4)$$

$$c_t = f_t^L c_{t-1}^L + f_t^R c_{t-1}^R + i_t \tanh(x_t) \quad (5)$$

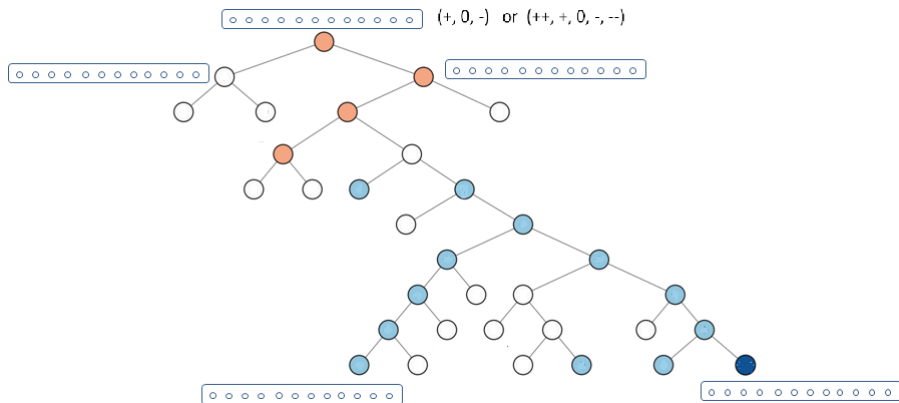
$$o_t = \sigma(W_{ho}^L h_{t-1}^L + W_{ho}^R h_{t-1}^R + W_{co} c_t + b_o) \quad (6)$$

$$h_t = o_t \tanh(c_t) \quad (7)$$

- If you have a non-binary tree, a simple solution is to binarize it.

Tree-LSTM Application: Sentiment Analysis

Sentiment composed over a constituency parse tree:



Tree-LSTM Application: Sentiment Analysis

Results on Stanford Sentiment Treebank (Zhu et al., 2015b):

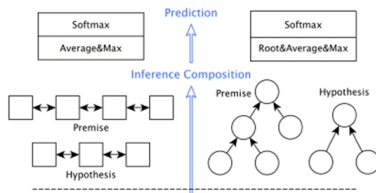
Models	roots	phrases
NB	41.0	67.2
SVM	40.7	64.3
RvNN	43.2	79.0
RNTN	45.7	80.7
Tree-LSTM	48.9	81.9

Table: Performances (accuracy) of models on Stanford Sentiment Treebank, at the sentence level (roots) and the phrase level.

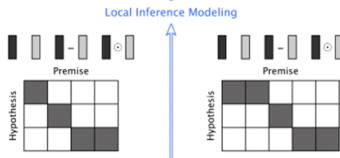
Tree-LSTM Application: Natural Language Inference

Applied to Natural Language Inference (NLI): Determine if a sentence entails another, if they contradict, or have no relation (Chen et al., 2017).

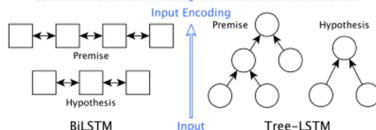
Inference
Composition



Local inference
modeling



Co-attention



Encoding input

Tree-LSTM Application: Natural Language Inference

Accuracy on Stanford Natural Language Inference (SNLI) dataset:
(Chen et al., 2017)

Model	#Para.	Train	Test
(1) Handcrafted features (Bowman et al., 2015)	-	99.7	78.2
(2) 300D LSTM encoders (Bowman et al., 2016)	3.0M	83.9	80.6
(3) 1024D pretrained GRU encoders (Vendrov et al., 2015)	15M	98.8	81.4
(4) 300D tree-based CNN encoders (Mou et al., 2016)	3.5M	83.3	82.1
(5) 300D SPINN-PI encoders (Bowman et al., 2016)	3.7M	89.2	83.2
(6) 600D BiLSTM intra-attention encoders (Liu et al., 2016)	2.8M	84.5	84.2
(7) 300D NSE encoders (Munkhdalai and Yu, 2016a)	3.0M	86.2	84.6
(8) 100D LSTM with attention (Rocktäschel et al., 2015)	250K	85.3	83.5
(9) 300D mLSTM (Wang and Jiang, 2016)	1.9M	92.0	86.1
(10) 450D LSTMN with deep attention fusion (Cheng et al., 2016)	3.4M	88.5	86.3
(11) 200D decomposable attention model (Parikh et al., 2016)	380K	89.5	86.3
(12) Intra-sentence attention + (11) (Parikh et al., 2016)	580K	90.5	86.8
(13) 300D NTI-SLSTM-LSTM (Munkhdalai and Yu, 2016b)	3.2M	88.5	87.3
(14) 300D re-read LSTM (Sha et al., 2016)	2.0M	90.7	87.5
(15) 300D btree-LSTM encoders (Paria et al., 2016)	2.0M	88.6	87.6
(16) 600D ESIM	4.3M	92.6	<u>88.0</u>
(17) HIM (600D ESIM + 300D Syntactic tree-LSTM)	7.7M	93.5	88.6

* Welcome to the poster at 6:00-9:30pm on July 31.

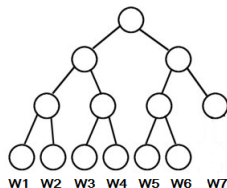
Learning Representation for Natural Language Inference

RepEval-2017 Shared Task (Williams et al., 2017): Learn sentence representation as a fixed-length vector.

Team	Matched Acc.	Mismatched Acc.
alpha (ensemble)	74.9%	74.9%
YixinNie-UNC-NLP	74.5%	73.5%
alpha	73.5%	73.6%
Rivercorners (ensemble)	72.2%	72.8%
Rivercorners	72.1%	72.1%
LCT-MALTA	70.7%	70.8%
TALP-UPC	67.9%	68.2%
BiLSTM baseline	67.0%	67.6%

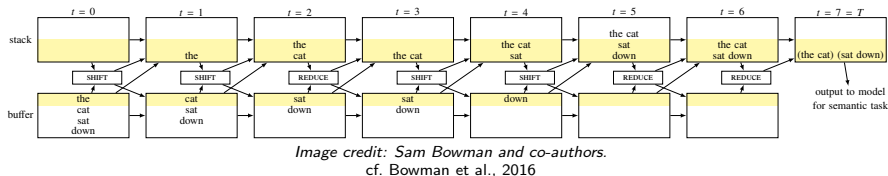
Tree-LSTM without Syntactic Parses

How if we simply apply recursive networks over trees that are not generated from syntactic parses, e.g., a *complete binary trees*?



- Multiple efforts on SNLI (Munkhdalai et al., 2016; Chen et al., 2017) have observed that the models outperform sequential (chain) LSTM.
- This could be related to the discussion that recursive nets may capture long-distance dependency (Goodfellow et al., 2016).

SPINN: Doing Away with Test-Time Trees



Shift-Reduce Parsers:

- Exploit isomorphism between binary branching trees with T leaves and sequences of $2T - 1$ binary shift/reduce actions.
- *Shift* unattached leaves from a buffer onto a processing stack.
- *Reduce* the top two child nodes on the stack to a single parent node.

SPINN: Jointly train a TreeRNN and a vector-based shift-reduce parser.

**Training time trees offer supervision for shift-reduce parser.
No need for test time trees!**

SPINN: Doing Away with Test-Time Trees

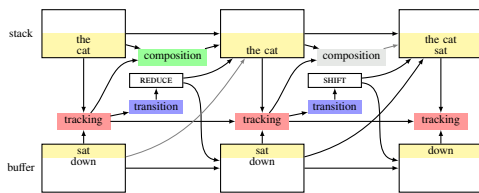


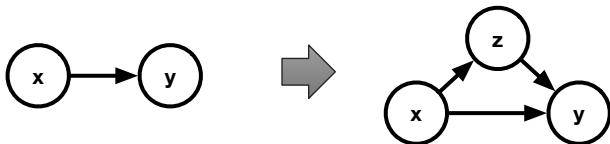
Image credit: Sam Bowman and co-authors.

- Word vectors start on buffer b (top: first word in sentence).
- Shift moves word vectors from buffer to stack s .
- Reduce pops top two vectors off the stack, applies $f^R : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$, and pushes the result back to the stack (i.e. TreeRNN composition).
- Tracker LSTM tracks parser/composer state across operations, decides shift-reduce operations a , is supervised by both observed shift-reduce operations and end-task:

$$h_t = LSTM(f^C(b_{t-1}[0], s_{t-1}[0], s_{t-1}[1]), h_{t-1}) \quad a_t \sim f^A(h_t)$$

A Quick Introduction to REINFORCE

What if some part of our process is not differentiable (e.g. samples from the shift-reduce module in SPINN) but we want to learn with no labels. . .



$$p(y|x) = \mathbb{E}_{p_{\theta}(z|x)} [f_{\phi}(z, x)] \quad \text{s.t. } y \sim f_{\phi}(z, x) \text{ or } y = f_{\phi}(z, x)$$

$$\nabla_{\phi} p(y|x) = \sum_z p_{\theta}(z|x) \nabla_{\phi} f_{\phi}(z, x) = \mathbb{E}_{p_{\theta}(z|x)} [\nabla_{\phi} f_{\phi}(z, x)]$$

$$\nabla_{\theta} p(y|x) = \sum_z f_{\phi}(z, x) \nabla_{\theta} p_{\theta}(z|x) = ???$$

A Quick Introduction to REINFORCE

The REINFORCE Trick (R. J. Williams, 1992)

$$\nabla_{\theta} \log p_{\theta}(z|x) = \frac{\nabla_{\theta} p_{\theta}(z|x)}{p_{\theta}(z|x)} \Rightarrow \nabla_{\theta} p_{\theta}(z|x) = p_{\theta}(z|x) \nabla_{\theta} \log p_{\theta}(z|x)$$

$$\begin{aligned} \nabla_{\theta} p(y|x) &= \sum_z f_{\phi}(z, x) \nabla_{\theta} p_{\theta}(z|x) \\ &= \sum_z f_{\phi}(z, x) p_{\theta}(z|x) \nabla_{\theta} \log p_{\theta}(z|x) \\ &= \mathbb{E}_{p_{\theta}(z|x)} [f_{\phi}(z, x) \nabla_{\theta} \log p_{\theta}(z|x)] \end{aligned}$$

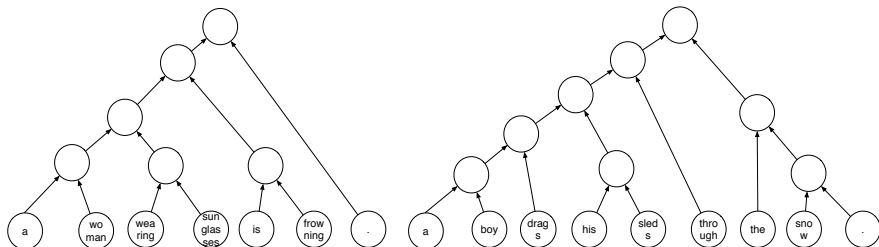
This naturally extends to cases where $p(z|x) = p(z_1, \dots, z_n|x)$.

RL vocab: samples of such sequences of discrete actions are referred to as “traces”. We often refer to $p_{\theta}(z|x)$ as a policy $\pi_{\theta}(z; x)$.

SPINN+RL: Doing Away with Training-Time Trees

“Drop in” extension to SPINN (Yogatama et al., 2016):

- Treat $a_t \sim f^A(h_t)$ as policy $\pi_\theta^A(a_t; h_t)$, trained via REINFORCE.
- Reward is negated loss of the end task, e.g. log-likelihood of the correct label.
- Everything else is trained by backpropagation against the end task: tracker LSTM, representations, etc. receive gradient both from the supervised objective, and from REINFORCE via the shift-reduce policy.

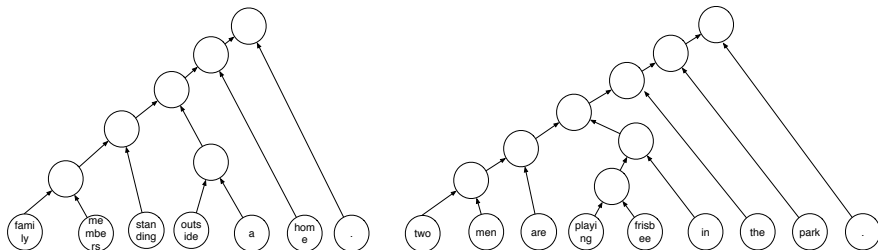


Model recovers linguistic-like structures (e.g. noun phrases, auxiliary verb-verb pairing, etc.).

SPINN+RL: Doing Away with Training-Time Trees

Does RL-SPINN work? According to Yogatama et al. (2016):

- Better than LSTM baselines: model captures and exploits structure.
- Better than SPINN benchmarks: model is not biased by what linguists think trees should be like, only has a loose inductive bias towards tree structures.
- But some parses do not reflect order of composition (see below).
Semi-supervised setup may be sensible.



Some "bad" parses, but not necessarily worse results.

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- **Convolutional composition models**
- Unsupervised models

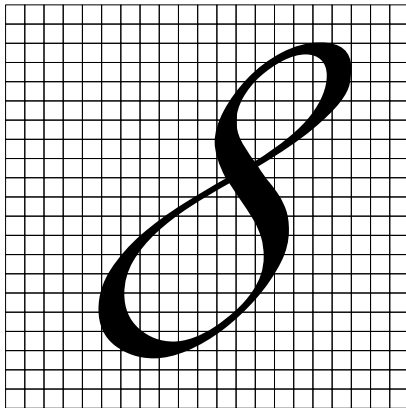
3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

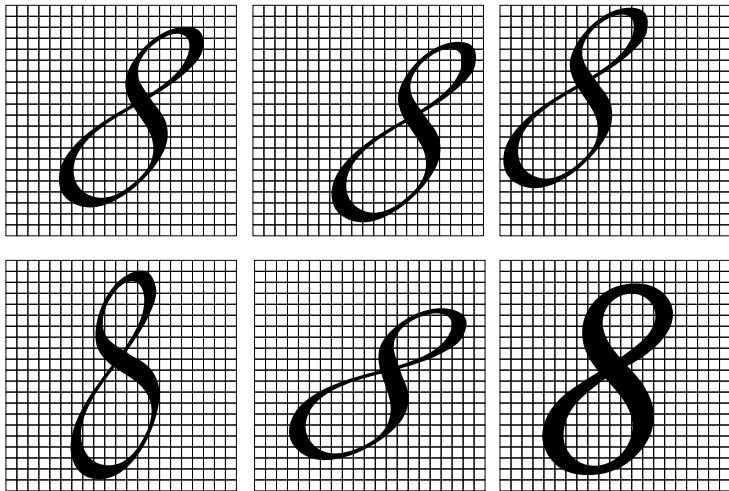
Convolution Neural Networks

Visual Inspiration: How do we learn to recognise pictures?
Will a fully connected neural network do the trick?



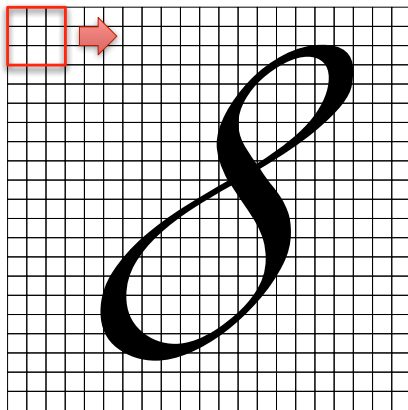
ConvNets for pictures

Problem: lots of variance that shouldn't matter (position, rotation, skew, difference in font/handwriting).



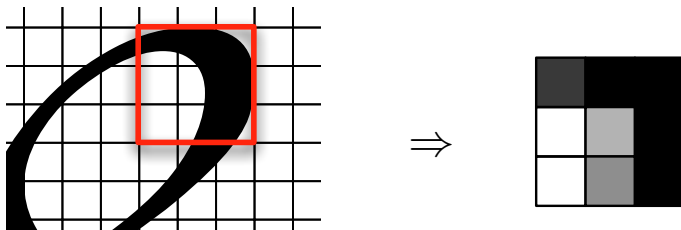
ConvNets for pictures

Solution: Accept that features are **local**. Search for local features with a window.



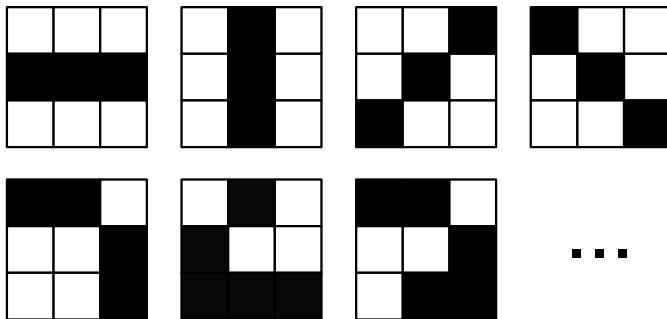
ConvNets for pictures

Convolutional window acts as a classifier for local features.



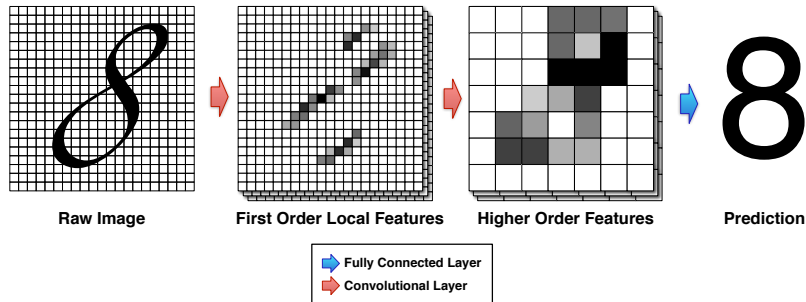
ConvNets for pictures

Different convolutional maps can be trained to recognise different features (e.g. edges, curves, serifs).



ConvNets for pictures

Stacked convolutional layers learn higher-level features.



One or more fully-connected layers learn classification function over highest level of representation.

ConvNets for language

Convolutional neural networks fit natural language well.

Deep ConvNets capture:

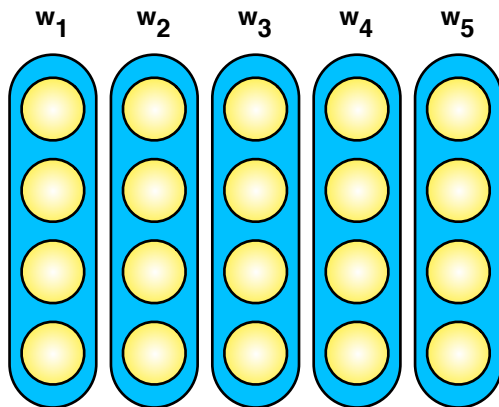
- Positional invariances
- Local features
- Hierarchical structure

Language has:

- Some positional invariance
- Local features (e.g. POS)
- Hierarchical structure (phrases, dependencies)

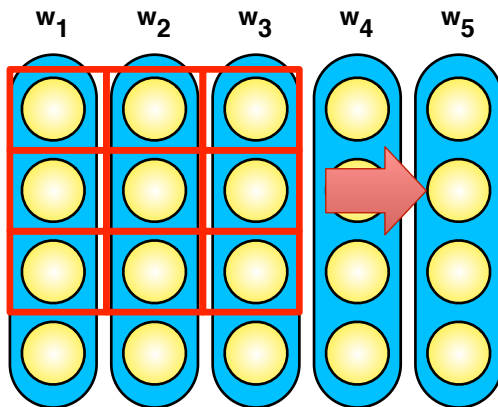
ConvNets for language

How do we go from images to sentences? Sentence matrices!



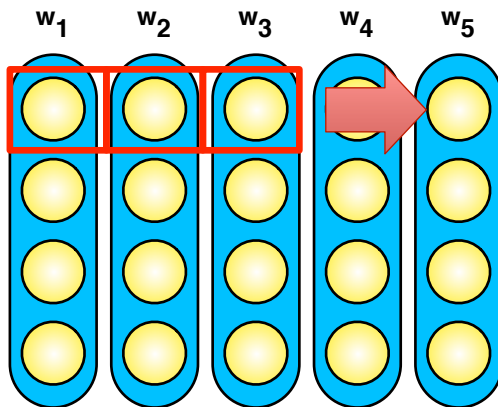
ConvNets for language

Does a convolutional window make sense for language?

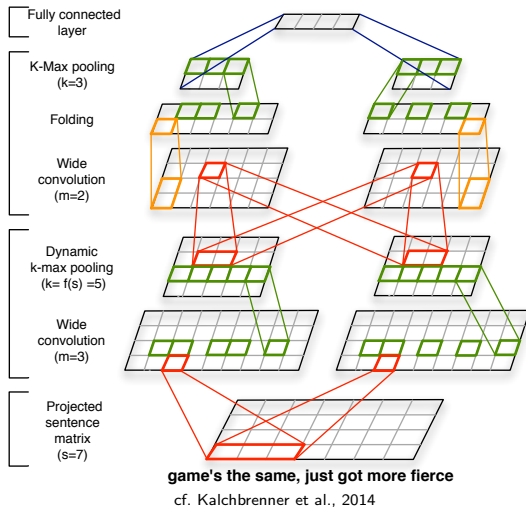


ConvNets for language

A better solution: feature-specific windows.



Word Level Sentence Vectors with ConvNets



Character Level Sentence Vectors with ConvNets

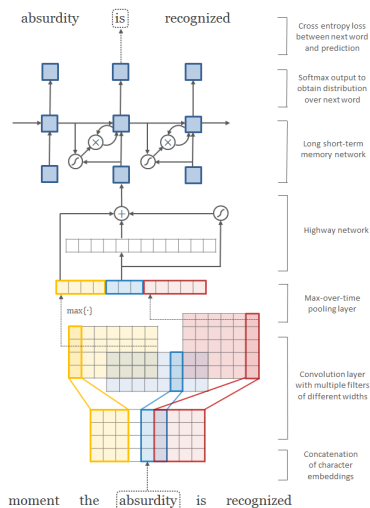


Image credit: Yoon Kim and co-authors.

cf. Kim et al., 2016

- Naively, we could just represent everything at character level.
- Convolutions seem to work well for low-level patterns (e.g. morphology)
- One interpretation: multiple filters can capture the low-level idiosyncrasies of natural language (e.g. arbitrary spelling) whereas language is more compositional at a higher level.

ConvNet-like Architectures for Composition

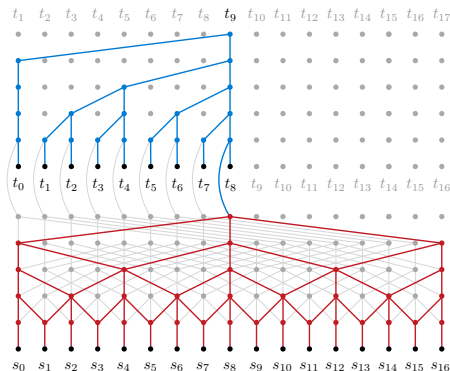


Image credit: Nal Kalchbrenner and co-authors.

cf. Kalchbrenner et al., 2016

- Many other CNN-like architectures (e.g. ByteNet from Kalchbrenner et al. (2016))
- Common recipe components: dilated convolutions and ResNet blocks.
- These model sequences well in domains like speech, and are beginning to find applications in NLP, so worth reading up on.

Outline

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- **Unsupervised models**

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

Unsupervised Composition Models

Why care about unsupervised learning?

- *Much* more unlabelled linguistic data than labelled data.
- Learn general purpose representations and composition functions.
- Suitable pre-training for supervised models, semi-supervised, or multi-task objectives.
- In the (paraphrased) words of Yann LeCun: unsupervised learning is a cake, supervised learning is frosting, and RL is the cherry on top!

Plot twist: it's possibly a cherry cake.

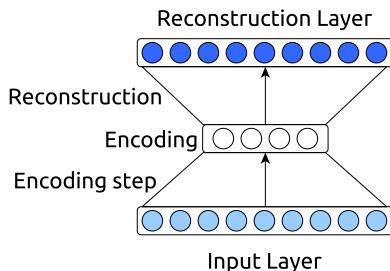
Yes, that's nice... But what are we doing, concretely?

Good question! Usually, just modelling—directly or indirectly—some aspect of the probability of the observed data.

Further suggestions on a postcard, please!

Autoencoders

Autoencoders provide an unsupervised method for representation learning:

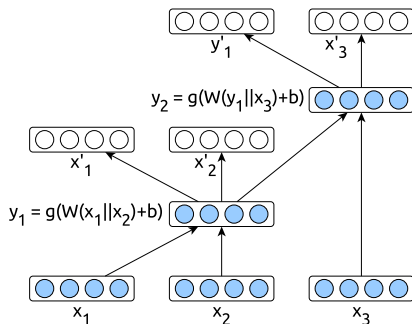


We minimise an objective function over inputs $x_i, i \in N$ and their reconstructions x'_i :

$$J = \frac{1}{2} \sum_i^N \|x'_i - x_i\|^2$$

Warning: degenerate solution if x_i can be updated ($\forall i. x_i = \mathbf{0}$).

Recursive Autoencoders



cf. Socher et al., 2011a

To auto-encode variable length sequences, we can chain autoencoders to create a recursive structure.

Objective Function

Minimizing the reconstruction error will learn a compression function over the inputs:

$$E_{rec}(i, \theta) = \frac{1}{2} \|x_i - x'_i\|^2$$

A “modern” alternative: use sequence to sequence model, and log-likelihood objective.

What's wrong with auto-encoders?

- Empirically, narrow auto-encoders produce sharp latent codes, and unregularised wide auto-encoders learn identity functions.
- Reconstruction objective includes nothing about distance preservation in latent space: no guarantee that

$$\begin{aligned} \text{dist}(a, b) &\leq \text{dist}(a, c) \\ \rightarrow \text{dist}(\text{encode}(a), \text{encode}(b)) &\leq \text{dist}(\text{encode}(a), \text{encode}(c)) \end{aligned}$$

- Conversely, little incentive for similar latent codes to generate radically different (but semantically equivalent) observations.

Ultimately, compression \neq meaning.

Skip-Thought

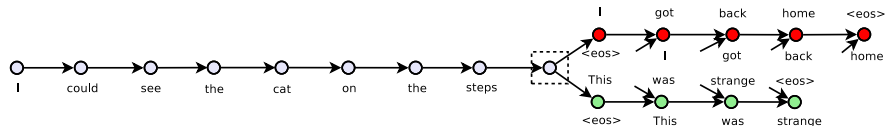


Image credit: Jamie Kiros and co-authors.
cf. Kiros et al., 2015

- Similar to auto-encoding objective: encode sentence, but decode *neighbouring sentences*.
- Pair of LSTM-based seq2seq models with share encoder, but alternative formulations are possible.
- Conceptually similar to distributional semantics: a unit's representation is a function of its neighbouring units, except units are sentence instead of words.

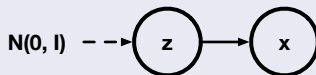
Variational Auto-Encoders

Semantically Weak Codes

Generally, auto-encoders sparsely encode or densely compress information. No pressure to ensure similarity continuum amongst codes.

Factorized Generative Picture

$$\begin{aligned} p(x) &= \int p(x, z) dz \\ &= \int p(x|z) p(z) dz \\ &= \mathbb{E}_{p(z)} [p(x|z)] \end{aligned}$$



Prior on z enforces semantic continuum (e.g. no arbitrarily unrelated codes for similar data), but expectation is typically intractable to compute exactly, and Monte Carlo estimate of gradients will be high variance.

Goal

Estimate, by maximising $p(x)$:

- The parameters θ of a function modelling part of the generative process $p_{\theta}(x|z)$ given samples from a fixed prior $z \sim p(z)$.
- The parameters ϕ of a distribution $q_{\phi}(z|x)$ approximating the true posterior $p(z|x)$.

How do we do it? We maximise $p(x)$ via a variational lower bound (VLB):

$$\log p(x) \geq \mathbb{E}_{q_{\phi}(z|x)} [\log p_{\theta}(x|z)] - D_{KL}(q_{\phi}(z|x) \| p(z))$$

Equivalently we can minimise $NLL(x)$:

$$NLL(x) \leq \mathbb{E}_{q_{\phi}(z|x)} [NLL_{\theta}(x|z)] + D_{KL}(q_{\phi}(z|x) \| p(z))$$

Variational Auto-Encoders

Let's derive the VLB:

$$\begin{aligned}\log p(x) &= \log \int \mathbf{1} \cdot p_{\theta}(x|z)p(z)dz \\ &= \log \int \frac{q_{\phi}(z|x)}{q_{\phi}(z|x)} p_{\theta}(x|z)p(z)dz \\ &= \log \mathbb{E}_{q_{\phi}(z|x)} \left[\frac{p(z)}{q_{\phi}(z|x)} p_{\theta}(x|z) \right] \\ &\geq \mathbb{E}_{q_{\phi}(z|x)} \left[\log \frac{p(z)}{q_{\phi}(z|x)} + \log p_{\theta}(x|z) \right] \\ &= \mathbb{E}_{q_{\phi}(z|x)} [\log p_{\theta}(x|z)] - D_{KL}(q_{\phi}(z|x) \| p(z))\end{aligned}$$

For right $q_{\phi}(z|x)$ and $p(z)$ (e.g. Gaussians) there is a closed-form expression of $D_{KL}(q_{\phi}(z|x) \| p(z))$.

Variational Auto-Encoders

The problem of stochastic gradients

Estimating $\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(z|x)} [\log p_{\theta}(x|z)]$ requires backpropagating through samples $z \sim q_{\phi}(z|x)$. For some choices of q , such as Gaussians there are *reparameterization tricks* (cf. Kingma et al., 2013)

Reparameterizing Gaussians (Kingma et al., 2013)

$$z \sim N(z; \mu, \sigma^2)$$

equivalent to $z = \mu + \sigma \epsilon$

where $\epsilon \sim N(\epsilon; \mathbf{0}, \mathbf{I})$

$$\text{Trivially: } \frac{\partial z}{\partial \mu} = 1 \quad \frac{\partial z}{\partial \sigma} = \epsilon$$

Variational Auto-Encoders for Sentences

- 1 Observe a sentence w_1, \dots, w_n . Encode it, e.g. with an LSTM:
 $h^e = LSTM^e(w_1, \dots, w_n)$
- 2 Predict $\mu = f^\mu(h^e)$ and $\sigma^2 = f^\sigma(h^e)$ (in practice we operate in log space for σ^2 by determining $\log \sigma$).
- 3 Sample $z \sim q(z|x) = N(z; \mu, \sigma^2)$
- 4 Use conditional RNN to decode and measure $\log p(x|z)$. Use closed-form formula of KL divergence of two Gaussians to calculate $-D_{KL}(q_\phi(z|x) \| p(z))$. Add both to obtain maximisation objective.
- 5 Backpropagate gradient through decoder normally based on log component of the objective, and use reparameterisation trick to backpropagate through sampling operation back to encoder.
- 6 Gradient of the KL divergence component of the loss with regard to the encoder parameters is straightforward backpropagation.

Variational Auto-Encoders and Autoregressivity

The problem of powerful auto-regressive decoders

We want to minimise $NLL(x) \leq \mathbb{E}_{q(z|x)}[NLL(x|z)] + D_{KL}(q(z|x)||p(z))$.
What if the decoder is powerful enough to model x without using z ?

A degenerate solution:

- If z can be ignored when minimising the reconstruction loss of x given z , the model can safely let $q(z|x)$ collapse to the prior $p(z)$ to minimise $D_{KL}(q(z|x)||p(z))$.
- Since q need not depend on x (e.g. the encoder can just ignore x and predict the mean and variance of the prior), z bears no relation to x .
- Result: useless encoder, useless latent variable.

Is this really a problem?

If your decoder is not auto-regressive (e.g. MLPs expressing the probability of pixels which are conditionally independent given z), then no.

If your decoder is an RNN and domain has systematic patterns, then yes.

Variational Auto-Encoders and Autoregressivity

What are some solutions to this problem?

- Pick a non-autoregressive decoder. If you care more about the latent code than having a good generative model (e.g. document modelling), this isn't a bad idea, but frustrating if this is the only solution.
- KL Annealing: set $\mathbb{E}_{q(z|x)}[NLL(x|z)] + \alpha D_{KL}(q(z|x)||p(z))$ as objective. Start with $\alpha = 0$ (basic seq2seq model). Increase α to 1 over time during training. Works somewhat, but unprincipled changing of the objective function.
- Set as objective $\mathbb{E}_{q(z|x)}[NLL(x|z)] + \max(\lambda, D_{KL}(q(z|x)||p(z)))$ where $\lambda \geq 0$ is a scalar or vector hyperparameter. Once the KL dips below λ , there is no benefit, so the model must rely on z to some extent. This objective is still a valid upper bound on $NLL(x)$ (albeit a looser one).

Outline

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

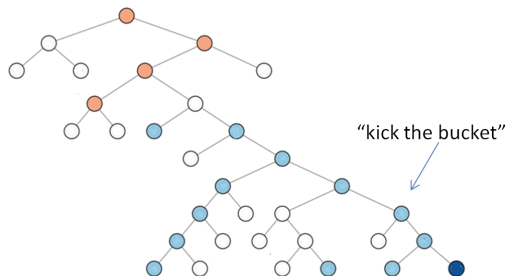
- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

Compositional or Non-compositional Representation



Compositional and Non-compositional Semantics

- Compositionality/non-compositionality is a common phenomenon in language.
- A framework that is able to consider both compositionality/non-compositionality is of interest.

Compositional and Non-compositional Semantics

- Compositionality/non-compositionality is a common phenomenon in language.
- A framework that is able to consider both compositionality/non-compositionality is of interest.
- A pragmatic viewpoint: If one is able to obtain holistically the representation of an n-gram or a phrase in text, it would be desirable that a composition model has the ability to decide the sources of knowledge it will use.

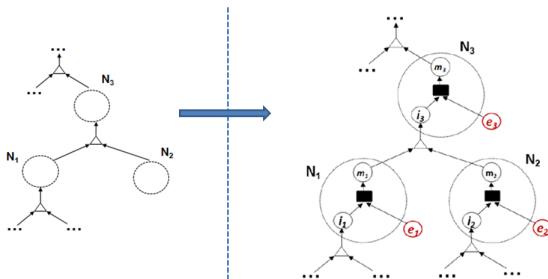
Compositional and Non-compositional Semantics

- Compositionality/non-compositionality is a common phenomenon in language.
- A framework that is able to consider both compositionality/non-compositionality is of interest.
- A pragmatic viewpoint: If one is able to obtain holistically the representation of an n-gram or a phrase in text, it would be desirable that a composition model has the ability to decide the sources of knowledge it will use.
- In addition to composition, considering non-compositionality may avoid back-propagating errors unnecessarily to confuse word embedding.
 - think about the “kick the bucket” example.

Integrating Compositional and Non-compositional Semantics

Integrating non-compositionality in recursive networks (Zhu et al., 2015a):

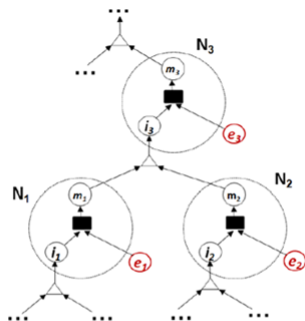
- Basic idea: Enabling individual composition operations to be able to choose information from different resources, compositional or non-compositional (e.g., holistically learned).



A framework for considering compositionality and non-compositionality in composition.

Integrating Compositional and Non-compositional Semantics

Model 1: Regular bilinear merge (Zhu et al., 2015a):

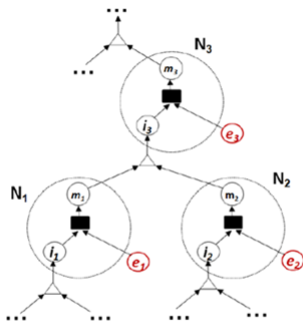


$$m_3 = \tanh(W_m \begin{bmatrix} i_3 \\ e_3 \end{bmatrix} + b_m)$$

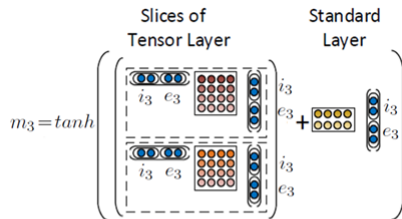
A framework for considering compositionality and non-compositionality in composition.

Integrating Compositional and Non-compositional Semantics

Model 2: Tensor-based merging (Zhu et al., 2015a)



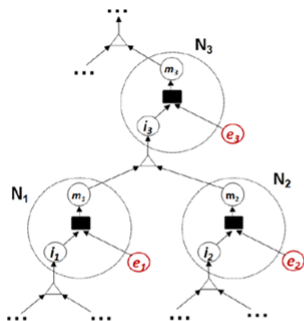
A framework for considering compositionality and non-compositionality in composition.



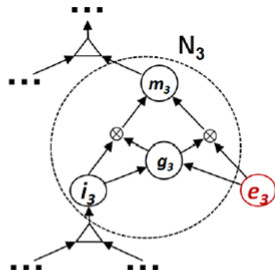
$$m_3 = \tanh\left(\begin{bmatrix} i_3 \\ e_3 \end{bmatrix}^T V_m^{[1:d]} \begin{bmatrix} i_3 \\ e_3 \end{bmatrix} + W_m \begin{bmatrix} i_3 \\ e_3 \end{bmatrix}\right)$$

Integrating Compositional and Non-compositional Semantics

Model 3: Explicitly gated merging (Zhu et al., 2015a):



A framework for considering compositionality and non-compositionality in composition.



$$g_3 = \sigma \left(\begin{bmatrix} W_{ge} e_3 \\ W_{gi} i_3 \end{bmatrix} + b_g \right)$$

$$m_3 = \tanh(W_m(g_3 \otimes \begin{bmatrix} i_3 \\ e_3 \end{bmatrix})) + b_m$$

Experiment Set-Up

- Task: sentiment analysis
- Data: Stanford Sentiment Treebank
- Non-compositional sentiment
 - Sentiment of ngrams automatically learned from tweets (Mohammad et al., 2013).
 - Polled the Twitter API every four hours from April to December 2012 in search of tweets with either a positive word hashtag or a negative word hashtag.
 - Using 78 seed hashtags (32 positive and 36 negative) such as #good, #excellent, and #terrible to annotate sentiment.
 - 775,000 tweets that contain at least a positive hashtag or a negative hashtag were used as the learning corpus.
 - Point-wise mutual information (PMI) is calculated for each bigrams and trigrams.
 - Each sentiment score is converted to a one-hot vector; e.g. a bigram with a score of -1.5 will be assigned a 5-dimensional vector [0, 1, 0, 0, 0] (i.e., the e vector).
 - Using the human annotation coming with Stanford Sentiment Treebank for bigrams and trigrams.

Results

Models	sentence-level (roots)	all phrases (all nodes)
(1) RNTN	42.44	79.95
(2) Regular-bilinear (auto)	42.37	79.97
(3) Regular-bilinear (manu)	42.98	80.14
(4) Explicitly-gated (auto)	42.58	80.06
(5) Explicitly-gated (manu)	43.21	80.21
(6) Confined-tensor (auto)	42.99	80.49
(7) Confined-tensor (manu)	43.75[†]	80.66[†]

Table: Model performances (accuracy) on predicting 5-category sentiment at the sentence (root) level and phrase level.

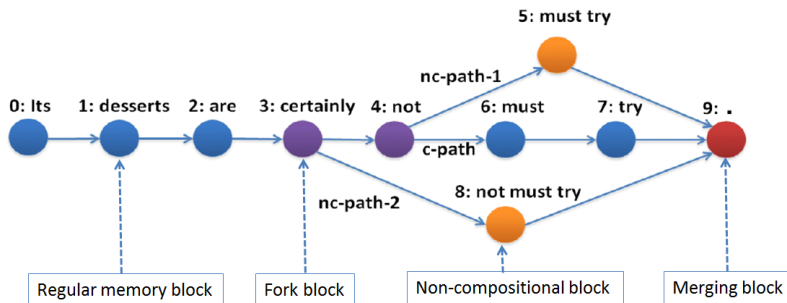
¹The results is based on the version 3.3.0 of the Stanford CoreNLP.

Integrating Compositional and Non-compositional Semantics

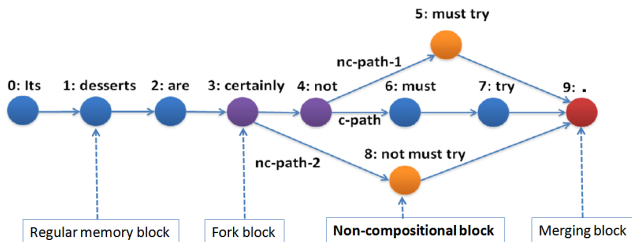
- We have discussed integrating non-compositionality in recursive networks.
- How if there are no prior input structures available?
 - Remember we have discussed the models that capture hidden structures.
- How if a syntactic parsing tree is not very reliable?
 - e.g., for data like social media text or speech transcripts.
- In these situations, how can we still consider non-compositionality in the composition process.

Integrating Compositional and Non-compositional Semantics

Integrating non-compositionality in chain recurrent networks (Zhu et al., 2016)



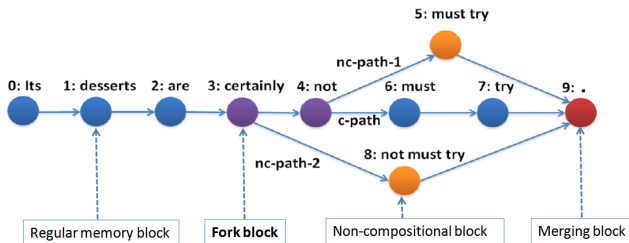
Integrating Compositional and Non-compositional Semantics



Non-compositional nodes:

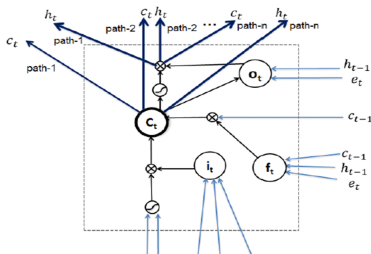
- Form the non-compositional paths (e.g., 3-8-9 or 4-5-9).
- Allow the embedding spaces of a non-compositional node to be different from those of a compositional node.

Integrating Compositional and Non-compositional Semantics

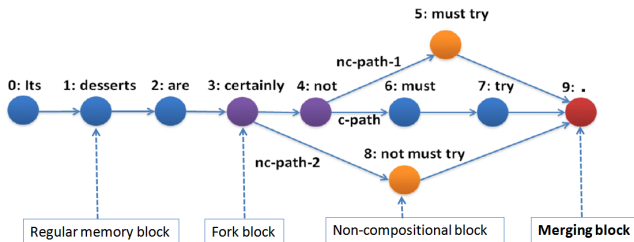


Fork nodes:

- Summarizing history so far to support both compositional and non-compositional paths.



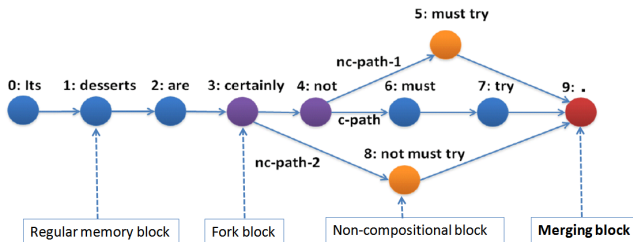
Integrating Compositional and Non-compositional Semantics



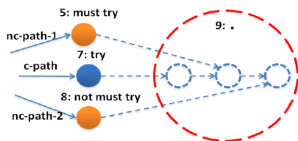
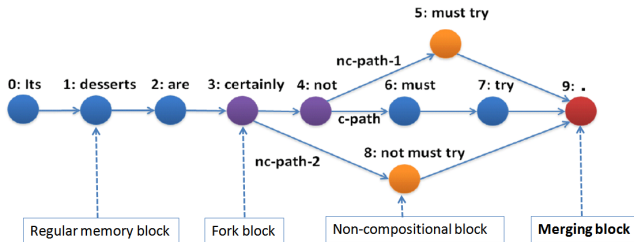
Merging nodes:

- Combining information from compositional and non-compositional paths.
- Binarization

Integrating Compositional and Non-compositional Semantics



Integrating Compositional and Non-compositional Semantics



Binarization:

- Binarizing the composition of in-bound paths (we do not worry too much about the order of merging.)
- Now we do not need to design different nodes for different fan-in, but let parameter-sharing be all over the nets.

Method	SemEval-13	SemEval-14
Majority baseline	29.19	34.46
Unigram (SVM)	56.95	58.58
3 rd best model	64.86	69.95
2 nd best model	65.27	70.14
The best model	69.02	70.96
DAG-LSTM	70.88	71.97

Table: Performances of different models in official evaluation metric (macro F-scores) on the test sets of SemEval-2013 and SemEval-2014 Sentiment Analysis in Twitter in predicting the sentiment of the tweet messages.

Method	SemEval-13	SemEval-14
DAG-LSTM		
Full paths	70.88	71.97
Full – {autoPaths}	69.36	69.27
Full – {triPaths}	70.16	70.77
Full – {triPaths, biPaths}	69.55	69.93
Full – {manuPaths}	69.88	70.58
LSTM without DAG		
Full – {autoPaths, manuPaths}	64.00	66.40

Table: Ablation performances (macro-averaged F-scores) of DAG-LSTM with different types of paths being removed.

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

Subword Composition

- Composition can also be performed to learn representations for words from subword components (Botha et al., 2014; Ling et al., 2015; Luong et al., 2015; Kim et al., 2016; Sennrich et al., 2016).
 - Rich morphology: some languages have larger vocabularies than others.
 - Informal text: *very coooooooooo!*

Subword Composition

- Composition can also be performed to learn representations for words from subword components (Botha et al., 2014; Ling et al., 2015; Luong et al., 2015; Kim et al., 2016; Sennrich et al., 2016).
 - Rich morphology: some languages have larger vocabularies than others.
 - Informal text: *very coooooooooo!*
Basically alleviate **Sparseness!**

Subword Composition

- Composition can also be performed to learn representations for words from subword components (Botha et al., 2014; Ling et al., 2015; Luong et al., 2015; Kim et al., 2016; Sennrich et al., 2016).
 - Rich morphology: some languages have larger vocabularies than others.
 - Informal text: *very coooooooooo!*
Basically alleviate **Sparseness!**
- One perspective of viewing subword models:
 - Morpheme based composition: deriving word representation from morphemes.
 - Character based composition: deriving word representation from characters (pretty effective as well, even used by itself!)

Subword Composition

- Composition can also be performed to learn representations for words from subword components (Botha et al., 2014; Ling et al., 2015; Luong et al., 2015; Kim et al., 2016; Sennrich et al., 2016).
 - Rich morphology: some languages have larger vocabularies than others.
 - Informal text: *very coooooooooo!*
Basically alleviate **Sparseness!**
- One perspective of viewing subword models:
 - Morpheme based composition: deriving word representation from morphemes.
 - Character based composition: deriving word representation from characters (pretty effective as well, even used by itself!)
- Another perspective (by model architectures):
 - Recursive models
 - Convolutional models
 - Recurrent models

Subword Composition

- Composition can also be performed to learn representations for words from subword components (Botha et al., 2014; Ling et al., 2015; Luong et al., 2015; Kim et al., 2016; Sennrich et al., 2016).
 - Rich morphology: some languages have larger vocabularies than others.
 - Informal text: *very coooooooooo!*
Basically alleviate **Sparseness!**
- One perspective of viewing subword models:
 - Morpheme based composition: deriving word representation from morphemes.
 - Character based composition: deriving word representation from characters (pretty effective as well, even used by itself!)
- Another perspective (by model architectures):
 - Recursive models
 - Convolutional models
 - Recurrent models
- We will discuss several typical methods here only briefly.

Subword Composition: Recursive Networks

Morphological Recursive Neural Networks (Luong et al., 2013):

Extending recursive neural networks (Socher et al., 2011b) to learn word representation through composition over morphemes.

- Assume the availability of morphemic analyses.
- Each tree node combines a stem vector and an affix vector.

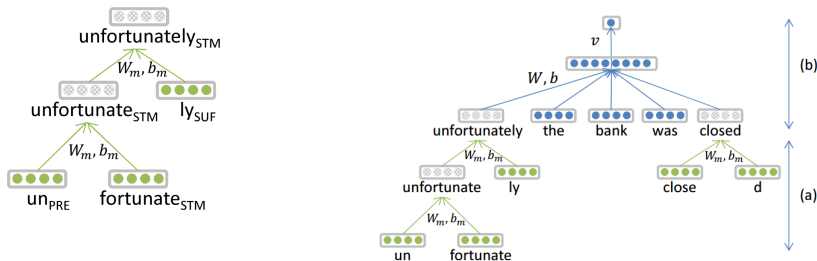


Figure. Context insensitive (left) and sensitive (right) Morphological Recursive Neural Networks.

Subword Composition: Recurrent Networks

Bi-directional LSTM for subword composition (Ling et al., 2015).

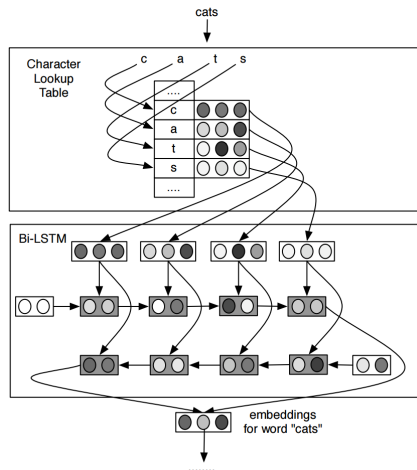


Figure. Character RNN for sub-word composition.

Subword Composition: Convolutional Networks

Convolutional neural networks for subword composition (Zhang et al., 2015)

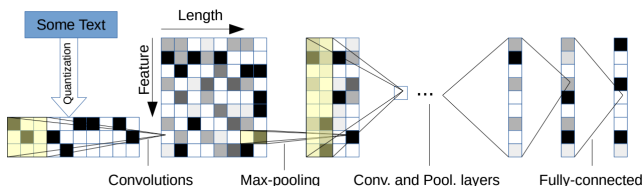


Figure. Character CNN for sub-word composition.

Subword Composition: Convolutional Networks

Convolutional neural networks for subword composition (Zhang et al., 2015)

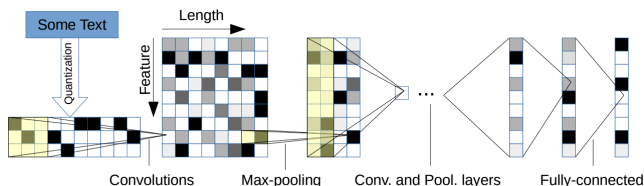


Figure. Character CNN for sub-word composition.

- In general, subword models have been successfully used in a wide variety of problems such as translation, sentiment analysis, question answering, etc.
- You should seriously consider it in the situations such as OOV is high or the word distribution has a long tail.

Outline

1 Introduction

- Semantic composition
- Formal methods
- Simple parametric models

2 Parameterizing Composition Functions

- Recurrent composition models
- Recursive composition models
- Convolutional composition models
- Unsupervised models

3 Selected Topics

- Compositionality and non-compositionality
- Subword composition methods

4 Summary

- The tutorial discusses **semantic composition** with **distributed representation** learned with neural networks.
- Neural networks are able to learn powerful representation and complicated composition functions.
- The models can achieve state-of-the-art performances on a wide range of NLP tasks.
- We expect further studies would continue to deepen our understanding on such approaches:
 - Unsupervised models
 - Compositionality with other “ingredients” of intelligence
 - Compositionality in multi-modalities
 - Interpretability of models
 - Distributed vs./and symbolic composition models
 -

References I



C. E. Osgood, G. J. Suci, and P. H. Tannenbaum. *The Measurement of Meaning*. University of Illinois Press, 1957.



Richard Montague. “English as a Formal Language”. In: *Linguaggi nella societa e nella tecnica*. Ed. by Bruno Visentini. Edizioni di Comunita, 1970, pp. 188–221.



G. A. Miller and P. N. Johnson-Laird. *Language and perception*. Cambridge, MA: Belknap Press, 1976.



J. A. Fodor and Z. W. Pylyshyn. “Connectionism and cognitive architecture: A critical analysis”. In: *Cognition* 28 (1988), pp. 3–71.



Jordan B. Pollack. “Recursive Distributed Representations”. In: *Artif. Intell.* 46.1-2 (1990), pp. 77–105.

References II



Ronald J. Williams. “Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning”. In: *Machine Learning* 8 (1992), pp. 229–256.



Barbara Partee. “Lexical semantics and compositionality”. In: *Invitation to Cognitive Science* 1 (1995), pp. 311–360.



Elie Bienenstock, Stuart Geman, and Daniel Potter. “Compositionality, MDL Priors, and Object Recognition”. In: *NIPS*. 1996.









Enrico Francesconi et al. “Logo Recognition by Recursive Neural Networks”. In: *GREC*. 1997.



Jeff Mitchell and Mirella Lapata. “Vector-based Models of Semantic Composition”. In: *ACL*. 2008, pp. 236–244.

References III

-  Richard Socher et al. “Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection”. In: *NIPS*. 2011, pp. 801–809.
-  Richard Socher et al. “Parsing Natural Scenes and Natural Language with Recursive Neural Networks”. In: *ICML*. 2011, pp. 129–136.
-  Richard Socher et al. “Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions”. In: *EMNLP*. 2011.
-  Richard Socher et al. “Semantic Compositionality through Recursive Matrix-Vector Spaces”. In: *EMNLP-CoNLL*. 2012, pp. 1201–1211.
-  Nal Kalchbrenner and Phil Blunsom. “Recurrent Continuous Translation Models.”. In: *EMNLP*. Vol. 3. 39. 2013, p. 413.
-  Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *CoRR* abs/1312.6114 (2013).

References IV



Thang Luong, Richard Socher, and Christopher D. Manning. “Better Word Representations with Recursive Neural Networks for Morphology”. In: *CoNLL*. 2013.



Saif Mohammad, Svetlana Kiritchenko, and Xiao-Dan Zhu. “NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets”. In: *SemEval@NAACL-HLT*. 2013.



Richard Socher et al. “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank”. In: *EMNLP*. 2013, pp. 1631–1642.



Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (2014).



Jan A. Botha and Phil Blunsom. “Compositional Morphology for Word Representations and Language Modelling”. In: *ICML*. 2014.

References V



Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. “A convolutional neural network for modelling sentences”. In: *arXiv preprint arXiv:1404.2188* (2014).



Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to sequence learning with neural networks”. In: *Advances in neural information processing systems*. 2014, pp. 3104–3112.



Xiaodan Zhu et al. “An Empirical Study on the Effect of Negation Words on Sentiment”. In: *ACL*. 2014.



Ryan Kiros et al. “Skip-thought vectors”. In: *Advances in neural information processing systems*. 2015, pp. 3294–3302.



Phong Le and Willem Zuidema. “Compositional Distributional Semantics with Long Short Term Memory”. In: **SEM@NAACL-HLT*. 2015.

References VI

-  Wang Ling et al. “Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation”. In: 2015.
-  Thang Luong et al. “Addressing the Rare Word Problem in Neural Machine Translation”. In: *ACL*. 2015.
-  Sheng Kai Tai, Richard Socher, and D. Christopher Manning. “Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks”. In: *ACL*. 2015, pp. 1556–1566.
-  Xiang Zhang and Yann LeCun. “Text Understanding from Scratch”. In: *CoRR* abs/1502.01710 (2015).
-  Xiaodan Zhu, Hongyu Guo, and Parinaz Sobhani. “Neural Networks for Integrating Compositional and Non-compositional Sentiment in Sentiment Composition”. In: **SEM@NAACL-HLT*. 2015.

References VII



Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. “Long Short-Term Memory Over Recursive Structures”. In: *ICML*. 2015, pp. 1604–1612.



Samuel R Bowman et al. “A fast unified model for parsing and sentence understanding”. In: *arXiv preprint arXiv:1603.06021* (2016).



Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.



Nal Kalchbrenner et al. “Neural Machine Translation in Linear Time”. In: *CoRR* abs/1610.10099 (2016).



Yoon Kim et al. “Character-Aware Neural Language Models”. In: *AAAI*. 2016.



B. M. Lake et al. “Building Machines that Learn and Think Like People”. In: *Behavioral and Brain Sciences*. (in press). (2016).

References VIII



Tsendsuren Munkhdalai and Hong Yu. “Neural Tree Indexers for Text Understanding”. In: *CoRR* abs/1607.04492 (2016).



Rico Sennrich, Barry Haddow, and Alexandra Birch. “Neural Machine Translation of Rare Words with Subword Units”. In: *ACL*. 2016.



Dani Yogatama et al. “Learning to Compose Words into Sentences with Reinforcement Learning”. In: *CoRR* abs/1611.09100 (2016).



Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. “DAG-Structured Long Short-Term Memory for Semantic Compositionality”. In: *NAACL*. 2016.



Qian Chen et al. “Enhanced LSTM for Natural Language Inference”. In: *ACL*. 2017.

References IX



Williams, Nikita Nangia, and Samuel R. Bowman. “A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference”. In: *CoRR* abs/1704.05426 (2017).