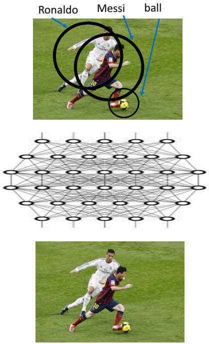
Inductive Bias of Deep Networks through Language Patterns

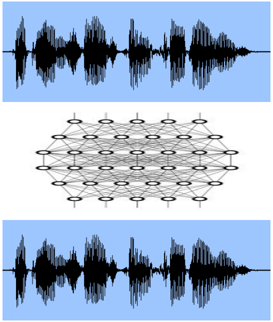
Roy Schwartz
University of Washington & Allen Institute for Artificial Intelligence

Joint work with *Yejin Choi*, **Ari Rappoport**, *Roi Reichart*, *Maarten Sap*, **Noah A. Smith** *and Sam Thomson*

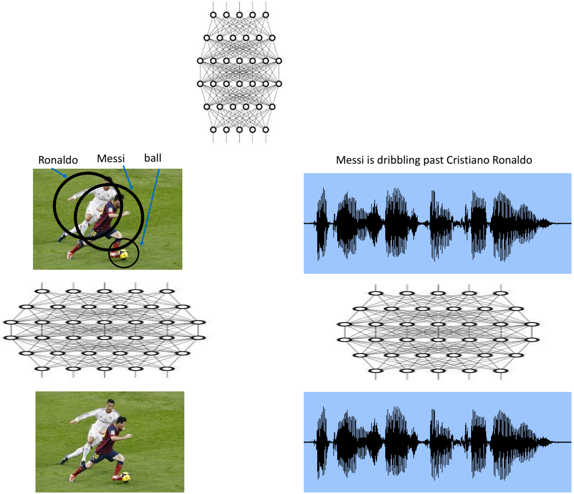
Google Research Tel-Aviv, December 21st, 2017



Messi is dribbling past Cristiano Ronaldo



What did Messi do?



Motivating Example

ROC Story Cloze Task (Mostafazadeh et al., 2016)

John and Mary have been dating for a while
Yesterday they had a date at a romantic restaurant
At one point John got down on his knees

Motivating Example

ROC Story Cloze Task (Mostafazadeh et al., 2016)

John and Mary have been dating for a while
Yesterday they had a date at a romantic restaurant
At one point John got down on his knees

Two competing endings:

- Option 1: John proposed
- Option 2: John tied his shoes

Motivating Example

ROC Story Cloze Task (Mostafazadeh et al., 2016)

John and Mary have been dating for a while
Yesterday they had a date at a romantic restaurant
At one point John got down on his knees

Two competing endings:

- Option 1: John proposed
- Option 2: John tied his shoes

A **hard** task

- One year after the release of the dataset, state-of-the-art was still $< 60\%$

Motivating Example—Inductive Bias

Schwartz et al., CoNLL 2017

- Our observation: the annotation of the dataset resulted in *writing biases*
  - E.g., *wrong* endings contain more negative terms
- Our solution: train a *pattern*-based classifier on the *endings only*
  - 72.5% accuracy on the task
- Combined with deep learning methods, we get 75.2% accuracy
  - First place in the LSDSem 2017 shared task

Outline

Case study 1: Word embeddings

Schwartz et al., CoNLL 2015, NAACL 2016

Case Study 2: Recurrent Neural Networks

Schwartz et al., in submission

Outline

Case study 1: Word embeddings

Schwartz et al., CoNLL 2015, NAACL 2016

Case Study 2: Recurrent Neural Networks

Schwartz et al., in submission

Distributional Semantics Models

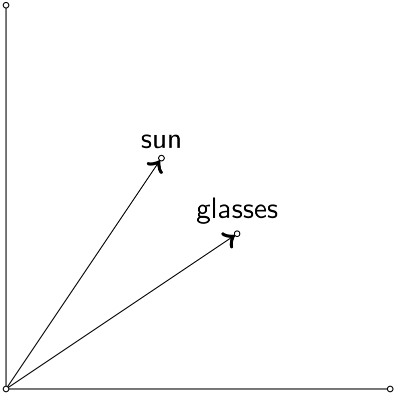
Aka, Vector Space Models, **Word Embeddings**

$$\mathbf{v}_{\text{sun}} = \begin{pmatrix} 0.23 \\ \text{-}0.21 \\ 0.15 \\ 0.61 \\ \vdots \\ 0.02 \\ \text{-}0.12 \end{pmatrix}, \mathbf{v}_{\text{glasses}} = \begin{pmatrix} 0.72 \\ 0.2 \\ 0.71 \\ 0.13 \\ \vdots \\ \text{-}0.1 \\ \text{-}0.11 \end{pmatrix}$$

Distributional Semantics Models

Aka, Vector Space Models, **Word Embeddings**

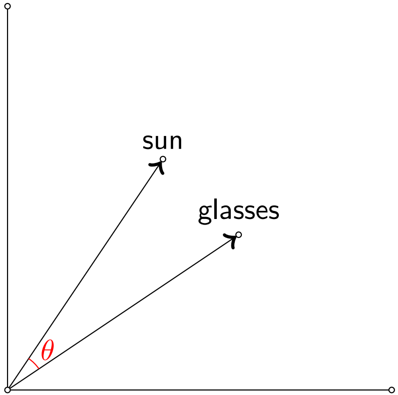
$$\mathbf{v}_{\text{sun}} = \begin{pmatrix} 0.23 \\ -0.21 \\ 0.15 \\ 0.61 \\ \vdots \\ 0.02 \\ -0.12 \end{pmatrix}, \mathbf{v}_{\text{glasses}} = \begin{pmatrix} 0.72 \\ 0.2 \\ 0.71 \\ 0.13 \\ \vdots \\ -0.1 \\ -0.11 \end{pmatrix}$$



Distributional Semantics Models

Aka, Vector Space Models, **Word Embeddings**

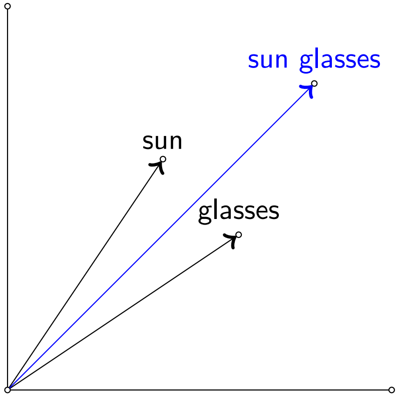
$$\mathbf{v}_{\mathsf{sun}} = \begin{pmatrix} 0.23 \\ -0.21 \\ 0.15 \\ 0.61 \\ \vdots \\ 0.02 \\ -0.12 \end{pmatrix}, \quad \mathbf{v}_{\mathsf{glasses}} = \begin{pmatrix} 0.72 \\ 0.2 \\ 0.71 \\ 0.13 \\ \vdots \\ -0.1 \\ -0.11 \end{pmatrix}$$



Distributional Semantics Models

Aka, Vector Space Models, **Word Embeddings**

$$\mathbf{v}_{\mathsf{sun}} = \begin{pmatrix} 0.23 \\ \text{-}0.21 \\ 0.15 \\ 0.61 \\ \vdots \\ 0.02 \\ \text{-}0.12 \end{pmatrix}, \mathbf{v}_{\mathsf{glasses}} = \begin{pmatrix} 0.72 \\ 0.2 \\ 0.71 \\ 0.13 \\ \vdots \\ \text{-}0.1 \\ \text{-}0.11 \end{pmatrix}$$



$V_{1.0}$: Count Models

Salton (1971)

- Each element $\mathbf{v}_{w_i} \in \mathbf{v}_w$ represents the *bag-of-words* co-occurrence of w with another word i in some text corpus
  - $\mathbf{v}_{\text{dog}} = (\text{cat: } 10, \text{ leash: } 15, \text{ loyal: } 27, \text{ bone: } 8, \text{ piano: } 0, \text{ cloud: } 0, \ldots)$
- Many variants of count models
  - Weighting schemes: PMI, TF-IDF
  - Dimensionality reduction: SVD/PCA

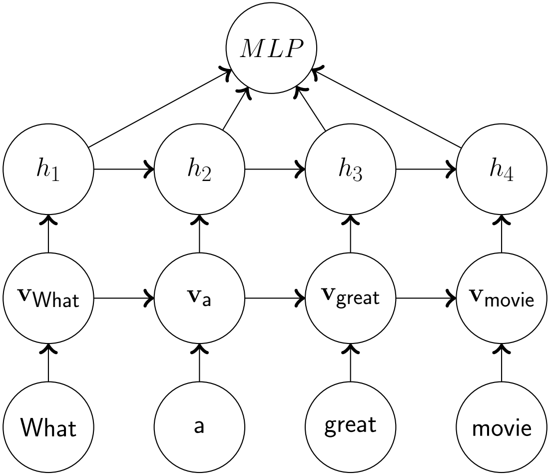
$V_{2.0}$: Predict Models

(Aka Word Embeddings; Bengio et al., 2003; Mikolov et al., 2013; Pennington et al., 2014)

- A new generation of vector space models
- Instead of representing vectors as cooccurrence counts, train a neural network to predict $p(\text{word}|\text{context})$
  - *context* is still defined as *bag-of-words* context
- Models learn a latent vector representation of each word
  - Developed to **initialize feature vectors** in deep learning models

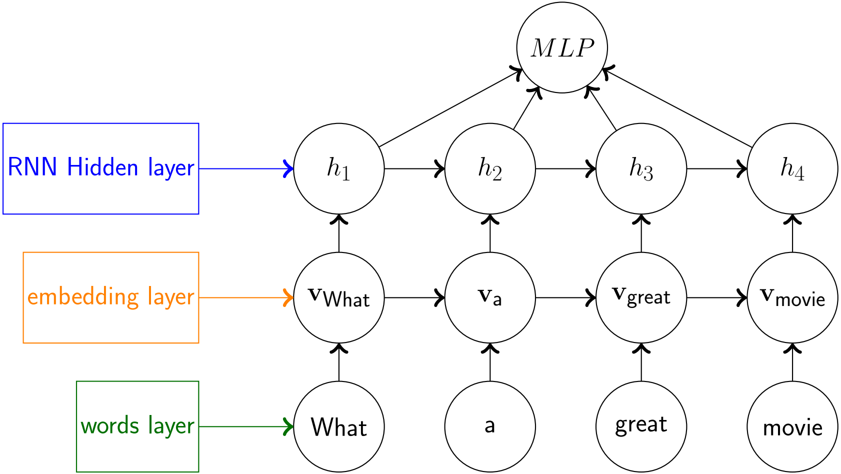
Recurrent Neural Networks

Elman (1990)



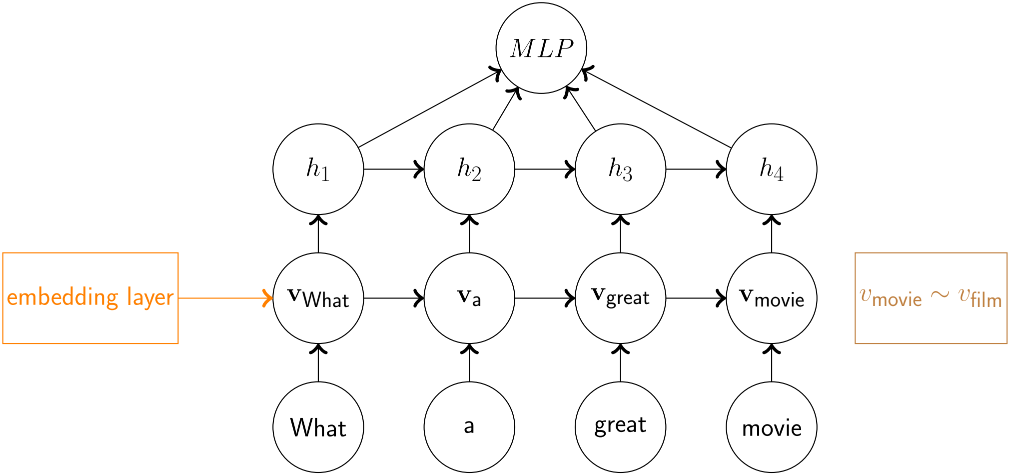
Recurrent Neural Networks

Elman (1990)



Recurrent Neural Networks

Elman (1990)



Word Embeddings — Problem

50 Shades of Similarity

- *Bag-of-word* contexts typically lead to **association** similarity
  - Captures general word association: coffee — cup, car — wheel
- Some applications prefer **functional** similarity
  - cup — glass, car — train
  - E.g., syntactic parsing

Symmetric Pattern Contexts

- *Symmetric patterns* are a special type of language patterns
  - *X and Y, X as well as Y*
- Words that appear in symmetric patterns are often *similar* rather than *related*
  - *read and write, smart as well as courageous*
  - **car and wheel, coffee as well as cup*
  - Davidov and Rappoport (2006); **Schwartz** et al. (2014)

Symmetric Pattern Example

I found the movie funny and enjoyable

- $\mathsf{c}_{BOW}(\text{funny}) = \{\text{I, found, the, } \textit{movie}\text{, and, enjoyable}\}$
- $\mathsf{c}_{BOW}(\text{movie}) = \{\text{I, found, the, } \textit{funny}\text{, and, enjoyable}\}$
- $\mathsf{c}_{symm\_patts}(\text{funny}) = \{\text{enjoyable}\}$
- $\mathsf{c}_{symm\_patts}(\text{movie}) = \{\}$

Solution: Inductive Bias using Symmetric Patterns

- Replace bag-of-words contexts with symmetric patterns
- Works both for count-based models and word embeddings
  - **Schwartz** et al. (2015; 2016)
- 5–20% performance increase on functional similarity tasks

Outline

Case study 1: Word embeddings

Schwartz et al., CoNLL 2015, NAACL 2016

Case Study 2: Recurrent Neural Networks

Schwartz et al., in submission

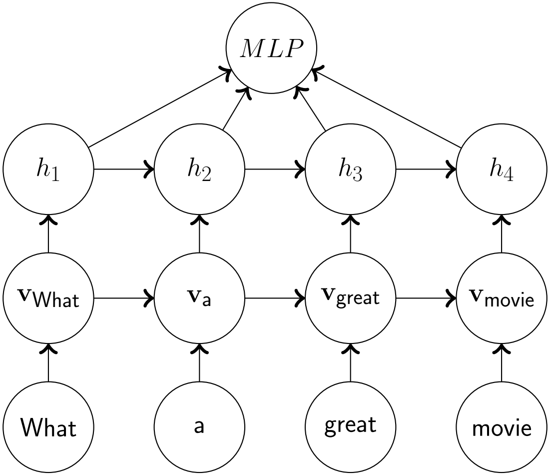
Recurrent Neural Networks

Elman (1990)

- RNNs are used as internal layers in deep networks
- Each RNN has a hidden state which is a function of both *the input* and the *previous hidden state*
- Variants of RNNs have become ubiquitous in NLP
  - In particular, long short-term memory (LSTM; Hochreiter and Schmidhuber, 1997) and gated recurrent unit (GRU; Cho et al., 2014)

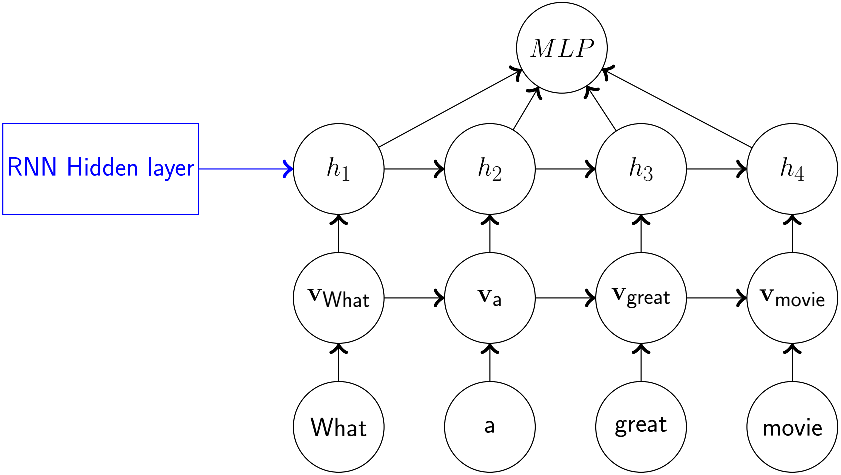
Recurrent Neural Networks

Elman (1990)



Recurrent Neural Networks

Elman (1990)



RNNs — Problems

- RNNs are heavily parameterizes, and thus prone to overfitting on *small datasets*
- RNNs are black boxes, and thus uninterpretable

Lexico-syntactic Patterns

Hard Patterns

- Patterns are sequences of words and wildcards (Hearst, 1992)
  - E.g., "X such as Y", "X was founded in Y", "what a great X!", "how big is the X?"
- Useful for many NLP tasks
- Information about the *words* filling the roles of the wildcards
  - **animals** such as **dogs**: **dog** is a type of an **animal**
  - **Google** was founded in **1998**
- Information about the *document*
  - what a great **movie**!: indication of a **positive** review

Flexible Patterns

Davidov et al. (2010)

Type	**Example**
`Exact match`	*What a great movie* !
`Inserted words`	*What a great* **funny** *movie* !
`Missing words`	*What great shoes* !
`Replaced words`	*What a* **wonderful** *book* !

Table: What a great X !

Flexible Patterns

Davidov et al. (2010)

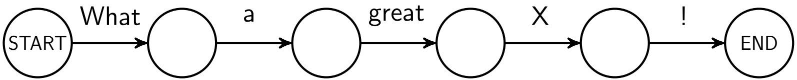
Type	**Example**
`Exact match`	*What a great movie* !
`Inserted words`	*What a great* **funny** *movie* !
`Missing words`	*What great shoes* !
`Replaced words`	*What a* **wonderful** *book* !

Table: What a great X !

- Can we go even *softer*?

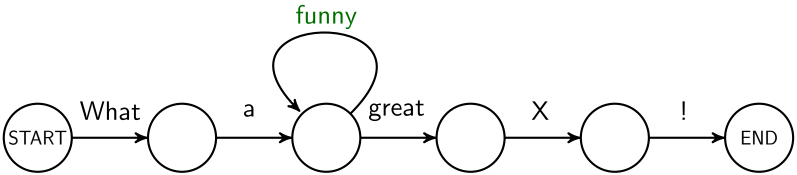
SoPa: An Interpretable Regular RNN

- We represent patterns as Weighted Finite State Automata with ϵ-transitions (ϵ-WSFA)
- A pattern P with d states over a vocabulary V is represented as a tuple $\langle \pi, T, \eta \rangle$
  - $\pi \in \mathbb{R}^d$ is an initial weight vector
  - $T \in (V \cup \{\epsilon\}) \to \mathbb{R}^{d \times d}$ is a transition weight function
  - $\eta \in \mathbb{R}^d$ is a final weight vector
- The score of a phrase $p_{\text{span}}(\boldsymbol{x}) = \pi^\top \mathbf{T}(\epsilon)^* \left(\prod_{i=1}^{n} \mathbf{T}(x_i)\mathbf{T}(\epsilon)^*\right) \eta$



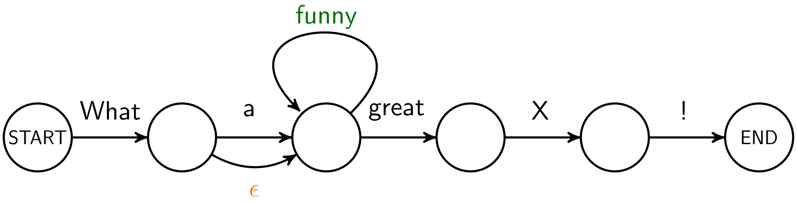
SoPa: An Interpretable Regular RNN

- We represent patterns as Weighted Finite State Automata with ϵ-transitions (ϵ-WSFA)
- A pattern P with d states over a vocabulary V is represented as a tuple $\langle \pi, T, \eta \rangle$
  - $\pi \in \mathbb{R}^d$ is an initial weight vector
  - $T \in (V \cup \{\epsilon\}) \to \mathbb{R}^{d \times d}$ is a transition weight function
  - $\eta \in \mathbb{R}^d$ is a final weight vector
- The score of a phrase $p_{\text{span}}(\boldsymbol{x}) = \pi^\top \mathbf{T}(\epsilon)^* \left(\prod_{i=1}^{n} \mathbf{T}(x_i)\mathbf{T}(\epsilon)^*\right) \eta$



SoPa: An Interpretable Regular RNN

- We represent patterns as Weighted Finite State Automata with ϵ-transitions (ϵ-WSFA)
- A pattern P with d states over a vocabulary V is represented as a tuple $\langle \pi, T, \eta \rangle$
  - $\pi \in \mathbb{R}^d$ is an initial weight vector
  - $T \in (V \cup \{\epsilon\}) \to \mathbb{R}^{d \times d}$ is a transition weight function
  - $\eta \in \mathbb{R}^d$ is a final weight vector
- The score of a phrase $p_{\text{span}}(\boldsymbol{x}) = \pi^\top \mathbf{T}(\epsilon)^* \left(\prod_{i=1}^{n} \mathbf{T}(x_i)\mathbf{T}(\epsilon)^*\right) \eta$



SoPa: *Soft* Patterns

- T is a parameterized function:

$$[\mathbf{T}(x)]_{i,j} = \begin{cases} \sigma(\mathbf{u}_i \cdot \mathbf{v}_x + a_i), & \text{if } j = i \textit{ (self-loop)} \\ \sigma(\mathbf{w}_i \cdot \mathbf{v}_x + b_i), & \text{if } j = i + 1 \textit{ (main path)} \\ 0, & \text{otherwise,} \end{cases} \tag{1a}$$

$$[\mathbf{T}(\epsilon)]_{i,j} = \begin{cases} \sigma(c_i), & \text{if } j = i + 1 \\ 0, & \text{otherwise,} \end{cases} \tag{1b}$$

  - x is a word, v_x is a pre-trained word embedding for x
  - w_i, u_i are vectors of parameters
  - a_i, b_i and c_i are scalar parameters

Concrete Word vs. Wildcards

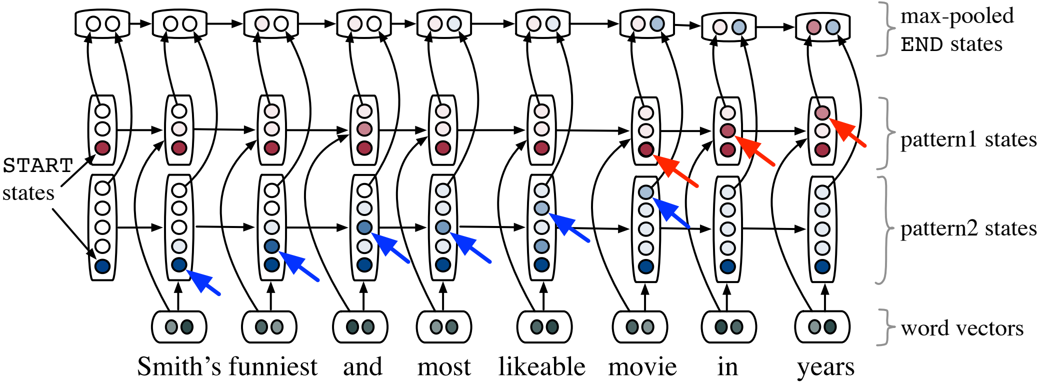
$$[\mathbf{T}(x)]_{i,j} = \begin{cases} \sigma(\mathbf{u}_i \cdot \mathbf{v}_x + a_i), & \text{if } j = i \textit{ (self-loop)} \\ \sigma(\mathbf{w}_i \cdot \mathbf{v}_x + b_i), & \text{if } j = i + 1 \textit{ (main path)} \\ 0, & \text{otherwise,} \end{cases}$$

- When $||w_i|| \approx 0$ and $b_i \gg 0$, $\mathbf{T}$ matches a wildcard
- When $||w_i|| \gg 0$, $b_i \ll 0$ and w_i is very close to a vector of some word (e.g., "what"), $\mathbf{T}$ is word specific
- Word embeddings allow $\mathbf{T}$ to "accept" classes of words (e.g., adjectives, concrete nouns, animate nouns)

Scoring a Document

- For a given pattern, compute the max of all matches in a document
  - The Viterbi algorithm (Viterbi, 1967)
- Randomly initialize k patterns, compute score for each one individually
  - This combination of k scores is the vector representation of the document
  - This representation is fed into a multilayer perceptron to classify a given document
- We keep a hidden state of the pattern matching along the document

SoPa as an RNN



SoPa: More Details

- We learn the patterns end-to-end
- We randomly initialize a set of 30–70 pattern WFSAs of varying lengths (2–7)
- Implementation in PyTorch
  - Adam optimizer, GloVe 840B embeddings, dropout
- Complexity: Assume k patterns, word embedding dimensionality v, maximum pattern length is d
  - The number of parameters in our model is $(2v + 3) \cdot d \cdot k$
  - For $k = 50$, $d = 6$, $v = 300$, this results in roughly $180K$ parameters

Experiments

- Three text classification datasets
- Baselines:
  - BiLSTM, DAN (Iyyer et al., 2015), Hard-patterns

RNNs — Problems

Reminder

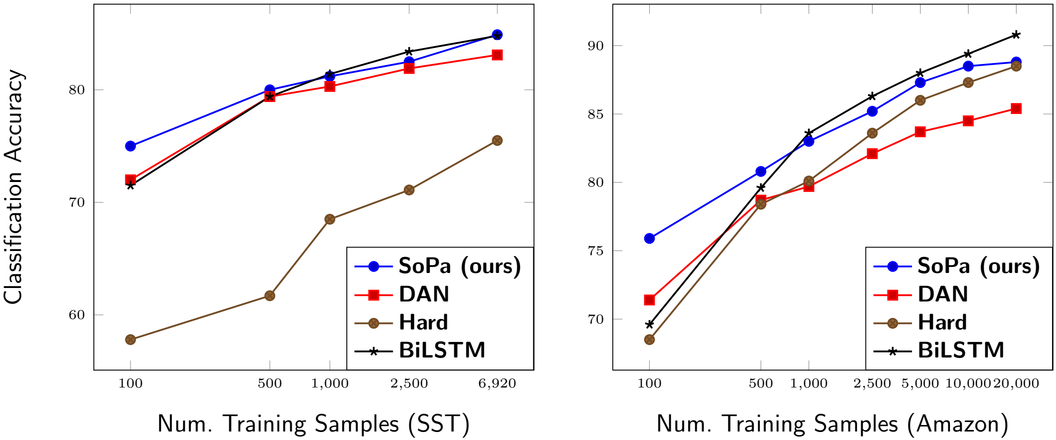
- RNNs are heavily parameterizes, and thus prone to overfitting on *small datasets*
- RNNs are black boxes, and thus uninterpretable

Results

Model	ROC	SST	Amazon
Hard	62.2% (4K)	75.5% (6K)	88.5% (67K)
DAN	64.3% (91K)	83.1% (91K)	85.4% (91K)
BiLSTM	65.2% (844K)	84.8% (1.5M)	90.8% (844K)
SoPa	64.9% (123K)	84.9% (255K)	88.8% (253K)

Results

Reduced Training Set



Interpretability

Individual Pattern

#States	Highest Scoring Phrases
6	thoughtful , reverent portrait of and astonishingly articulate cast of entertaining , thought-provoking film with gentle , mesmerizing portrait of poignant and uplifting story in
6	's uninspired story . this bad on purpose this leaden comedy . a half-assed film . is clumsy , the writing

#States	Highest Scoring Phrases
5	honest , and enjoyable soulful , scathing and joyous unpretentious , charming , quirky forceful , and beautifully energetic , and surprisingly
3	five minutes four minutes final minutes first half-hour fifteen minutes

Interpretability

Complete Document

Analyzed Documents
it's dumb, **but more importantly**, *it's just not scary*
though moonlight mile is replete with **acclaimed actors and actresses** and tackles a subject that's **potentially moving**, the movie is *too predictable* and *too self-conscious to reach a* level of **high drama**
While **its careful pace and** seemingly *opaque story* may not satisfy every moviegoer's appetite, the film's final scene is **soaringly, transparently moving**
unlike the speedy wham-bam effect *of most hollywood offerings*, character development – and more **importantly, character empathy – is at the heart of** italian for beginners.
the band's courage in the face of official repression **is inspiring**, **especially for** aging *hippies* (this one included).

Future Work

- Further improving SoPa
  - Loading pre-computed patterns
  - SoPa on top of BiLSTM
- Applying SoPa to other NLP tasks
  - Question answering, Text generation

Summary

- Deep learning is great!
  - But domain knowledge about language (**inductive bias**) is important to make it work well in practice
- Patterns are a particularly useful for source of inductive bias
  - Applications in word embeddings, RNNs, style detection

Summary

- Deep learning is great!
  - But domain knowledge about language (**inductive bias**) is important to make it work well in practice
- Patterns are a particularly useful for source of inductive bias
  - Applications in word embeddings, RNNs, style detection

Thank you!

Roy Schwartz homes.cs.washington.edu/~roysch/ roysch@cs.washington.edu

References I

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *JMLR*, 3:1137–1155, 2003.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *Proc. of SSST*, 2014.

Dmitry Davidov and Ari Rappoport. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *Proc. of ACL*, 2006.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. Enhanced sentiment learning using twitter hashtags and smileys. In *Proc. of COLING*, 2010.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. of ACL*, 1992.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proc. of ACL*, 2015.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of ICLR*, 2015.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013. arXiv:1301.3781.

References II

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proc. of NAACL*, 2016.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proc. of EMNLP*, 2014.

Gerard Salton. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971.

Roy Schwartz, Roi Reichart, and Ari Rappoport. Minimally supervised classification to semantic categories using automatically acquired symmetric patterns. In *Proc. of COLING*, 2014.

Roy Schwartz, Roi Reichart, and Ari Rappoport. Symmetric pattern based word embeddings for improved word similarity prediction. In *Proc. of CoNLL*, 2015.

Roy Schwartz, Roi Reichart, and Ari Rappoport. Symmetric patterns and coordinations: Fast and enhanced representations of verbs and adjectives. In *Proc. of NAACL*, 2016.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Li Zilles, Yejin Choi, and Noah A. Smith. The effect of different writing tasks on linguistic style: A case study of the ROC story cloze task. In *Proc. of CoNLL*, 2017.

A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, April 1967. ISSN 0018-9448. doi: 10.1109/TIT.1967.1054010.

Viterbi Recurrences

- Definitions:

$$[\mathrm{maxmul}(\mathbf{A}, \mathbf{B})]_{i,j} = \max_k A_{i,k} B_{k,j} \tag{2a}$$

$$\mathrm{eps}\left(\mathbf{v}\right) = \mathrm{maxmul}\left(\mathbf{v}, \max(\mathbf{I}, \mathbf{T}(\epsilon))\right) \tag{2b}$$

- Recurrences:

$$\mathbf{h}_0 = \mathrm{eps}(\pi^\top) \tag{3a}$$

$$\mathbf{h}_{t+1} = \max\left(\mathrm{eps}(\mathrm{maxmul}\left(\mathbf{h}_t, \mathbf{T}(x_t)\right)), \mathbf{h}_0\right) \tag{3b}$$

$$s_t = \mathrm{maxmul}\left(\mathbf{h}_t, \eta\right) \tag{3c}$$

$$s_{\mathsf{doc}} = \max_{0 \le t \le n} s_t \tag{3d}$$

Back to main

Self-loops and ϵ-transition

#States	Highest Scoring Phrases
6	thoughtful , reverent portrait of and astonishingly articulate cast of entertaining , thought-provoking film with gentle , mesmerizing portrait of poignant and uplifting story in
6	's ϵ uninspired story . this ϵ bad on purpose this ϵ leaden comedy . a ϵ half-assed film . is ϵ clumsy $,_{SL}$ the writing

#States	Highest Scoring Phrases
5	honest , and enjoyable soulful , scathing$_{SL}$ and joyous unpretentious , charming$_{SL}$, quirky forceful , and beautifully energetic , and surprisingly
3	five minutes four minutes final minutes first half-hour fifteen minutes

Back to main