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
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**

\[
v_{\text{sun}} = \begin{pmatrix} 0.23 \\ -0.21 \\ 0.15 \\ 0.61 \\ 0.02 \\ -0.12 \end{pmatrix}, \quad v_{\text{glasses}} = \begin{pmatrix} 0.72 \\ 0.2 \\ 0.71 \\ 0.13 \\ \vdots \\ -0.1 \end{pmatrix}
\]
Distributional Semantics Models

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

\[
v_{\text{sun}} = \begin{pmatrix}
0.23 \\
-0.21 \\
0.15 \\
0.61 \\
\vdots \\
0.02 \\
-0.12
\end{pmatrix},
\quad
v_{\text{glasses}} = \begin{pmatrix}
0.72 \\
0.2 \\
0.71 \\
0.13 \\
\vdots \\
-0.1 \\
-0.11
\end{pmatrix}
\]
Distributional Semantics Models
Aka, Vector Space Models, Word Embeddings

\[ \mathbf{v}_{\text{sun}} = \begin{pmatrix} 0.23 \\ -0.21 \\ 0.15 \\ 0.61 \\ 0.02 \\ -0.12 \end{pmatrix}, \quad \mathbf{v}_{\text{glasses}} = \begin{pmatrix} 0.72 \\ 0.2 \\ 0.71 \\ 0.13 \\ -0.1 \end{pmatrix} \]
Distributional Semantics Models
Aka, Vector Space Models, **Word Embeddings**

\[ \mathbf{v}_{\text{sun}} = \begin{pmatrix} 0.23 \\ -0.21 \\ 0.15 \\ 0.61 \\ 0.02 \\ -0.12 \end{pmatrix}, \quad \mathbf{v}_{\text{glasses}} = \begin{pmatrix} 0.72 \\ 0.2 \\ 0.71 \\ 0.13 \\ -0.1 \\ -0.11 \end{pmatrix} \]
Each element $v_{wi} \in v_w$ represents the *bag-of-words* co-occurrence of $w$ with another word $i$ in some text corpus

- $v_{\text{dog}} = (\text{cat}: 10, \text{leash}: 15, \text{loyal}: 27, \text{bone}: 8, \text{piano}: 0, \text{cloud}: 0, \ldots )$

Many variants of count models

- Weighting schemes: PMI, TF-IDF
- Dimensionality reduction: SVD/PCA
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)

What a great movie

h_1 \rightarrow h_2 \rightarrow h_3 \rightarrow h_4

\text{RNN Hidden layer}

v_{\text{What}} \rightarrow v_a \rightarrow v_{\text{great}} \rightarrow v_{\text{movie}}

\text{embedding layer}

\text{words layer}

MLP
Recurrent Neural Networks

Elman (1990)

What a great movie

$v_{\text{What}} \rightarrow v_a \rightarrow v_{\text{great}} \rightarrow v_{\text{movie}}$

$\text{What} \rightarrow a \rightarrow \text{great} \rightarrow \text{movie}$

$MLP$

$h_1 \rightarrow h_2 \rightarrow h_3 \rightarrow h_4$

embedding layer

$v_{\text{movie}} \sim v_{\text{film}}$
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

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

What a great movie

RNN Hidden layer

MLP

$h_1 \rightarrow h_2 \rightarrow h_3 \rightarrow h_4$

$v_{\text{What}} \rightarrow v_a \rightarrow v_{\text{great}} \rightarrow v_{\text{movie}}$

What a great movie
RNNs — Problems

- RNNs are heavily parameterized, 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)

<table>
<thead>
<tr>
<th>Type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact match</td>
<td><em>What a great movie!</em></td>
</tr>
<tr>
<td>Inserted words</td>
<td><em>What a great funny movie!</em></td>
</tr>
<tr>
<td>Missing words</td>
<td><em>What great shoes!</em></td>
</tr>
<tr>
<td>Replaced words</td>
<td><em>What a wonderful book!</em></td>
</tr>
</tbody>
</table>

**Table:** What a great X!
Flexible Patterns  
Davidov et al. (2010)

<table>
<thead>
<tr>
<th>Type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact match</td>
<td>What a great movie !</td>
</tr>
<tr>
<td>Inserted words</td>
<td>What a great <strong>funny</strong> movie !</td>
</tr>
<tr>
<td>Missing words</td>
<td>What great shoes !</td>
</tr>
<tr>
<td>Replaced words</td>
<td>What a <strong>wonderful</strong> book !</td>
</tr>
</tbody>
</table>

Table: What a great X !

- Can we go even softer?
SoPa: An Interpretable Regular RNN

- We represent patterns as Weighted Finite State Automata with $\epsilon$-transitions ($\epsilon$-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 $p_{\text{span}}(x) = \pi^\top T(\epsilon)^* (\prod_{i=1}^{n} T(x_i)T(\epsilon)^*) \eta$
SoPa: An Interpretable Regular RNN

- We represent patterns as Weighted Finite State Automata with $\epsilon$-transitions ($\epsilon$-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
  - $T \in (V \cup \{\epsilon\}) \rightarrow \mathbb{R}^{d \times d}$ is a transition weight function
  - $\eta \in \mathbb{R}^d$ is a final weight vector
- The score of a phrase $p_{\text{span}}(x) = \pi^\top T(\epsilon)^* (\prod_{i=1}^n T(x_i)T(\epsilon)^*) \eta$

![Diagram of a Weighted Finite State Automaton (WSFA)]

What a great funny X!
SoPa: An Interpretable Regular RNN

- We represent patterns as Weighted Finite State Automata with $\epsilon$-transitions ($\epsilon$-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
  - $T \in (V \cup \{\epsilon\}) \rightarrow \mathbb{R}^{d \times d}$ is a transition weight function
  - $\eta \in \mathbb{R}^d$ is a final weight vector
- The score of a phrase $p_{\text{span}}(x) = \pi^\top T(\epsilon)^* (\prod_{i=1}^n T(x_i)T(\epsilon)^*) \eta$

What a great funny X
SoPa: *Soft Patterns*

- \( T \) is a parameterized function:

\[
[T(x)]_{i,j} = \begin{cases} 
\sigma(u_i \cdot v_x + a_i), & \text{if } j = i \ (\text{self-loop}) \\
\sigma(w_i \cdot v_x + b_i), & \text{if } j = i + 1 \ (\text{main path}) \\
0, & \text{otherwise},
\end{cases}
\] (1a)

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

\[ [T(x)]_{i,j} = \begin{cases} 
\sigma(u_i \cdot v_x + a_i), & \text{if } j = i \ (self-loop) \\
\sigma(w_i \cdot v_x + b_i), & \text{if } j = i + 1 \ (main \ path) \\
0, & \text{otherwise,}
\end{cases} \]

- When \( ||w_i|| \approx 0 \) and \( b_i \gg 0 \), \( T \) 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”), \( T \) is word specific
- Word embeddings allow \( T \) 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

Smith’s funniest and most likeable movie in years

max-pooled END states

pattern1 states

pattern2 states

word vectors

START states

Smith’s funniest and most likeable movie in years

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 parameterized, and thus prone to overfitting on *small datasets*
- RNNs are black boxes, and thus uninterpretable
## Results

<table>
<thead>
<tr>
<th>Model</th>
<th>ROC</th>
<th>SST</th>
<th>Amazon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard</td>
<td>62.2% (4K)</td>
<td>75.5% (6K)</td>
<td>88.5% (67K)</td>
</tr>
<tr>
<td>DAN</td>
<td>64.3% (91K)</td>
<td>83.1% (91K)</td>
<td>85.4% (91K)</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>65.2% (844K)</td>
<td>84.8% (1.5M)</td>
<td>90.8% (844K)</td>
</tr>
<tr>
<td>SoPa</td>
<td>64.9% (123K)</td>
<td>84.9% (255K)</td>
<td>88.8% (253K)</td>
</tr>
</tbody>
</table>
Results

Reduced Training Set

![Graph showing classification accuracy vs. number of training samples for SST and Amazon datasets. The graph plots classification accuracy on the y-axis and number of training samples on the x-axis. The datasets are categorized into SoPa (ours), DAN, Hard, and BiLSTM. The graphs show trends in accuracy as the number of training samples increases.]
## Interpretability

### Individual Pattern

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

<table>
<thead>
<tr>
<th>States</th>
<th>Highest Scoring Phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>honest, and enjoyable soulful, scathing and joyous unpretentious, charming, quirky forceful, and beautifully energetic, and surprisingly</td>
</tr>
<tr>
<td>3</td>
<td>five minutes four minutes final minutes first half-hour fifteen minutes</td>
</tr>
</tbody>
</table>
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 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 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


Viterbi Recurrences

► Definitions:

\[
\text{[maxmul}(A, B)\text{]}_{i,j} = \max_k A_{i,k} B_{k,j} \quad (2a)
\]

\[
\text{eps}\left(v\right) = \text{maxmul}\left(v, \text{max}(I, T(\epsilon))\right) \quad (2b)
\]

► Recurrences:

\[
h_0 = \text{eps}\left(\pi^T\right) \quad (3a)
\]

\[
h_{t+1} = \max (\text{eps}(\text{maxmul}(h_t, T(x_t))), h_0) \quad (3b)
\]

\[
s_t = \text{maxmul}(h_t, \eta) \quad (3c)
\]

\[
s_{\text{doc}} = \max_{0 \leq t \leq n} s_t \quad (3d)
\]
## Self-loops and $\epsilon$-transition

<table>
<thead>
<tr>
<th>#States</th>
<th>Highest Scoring Phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>thoughtful, reverent portrait of and astonishingly articulate cast of entertaining, thought-provoking film with gentle, mesmerizing portrait of poignant and uplifting story in</td>
</tr>
<tr>
<td>6</td>
<td>'s $\epsilon$ uninspired story . this $\epsilon$ bad on purpose this $\epsilon$ leden comedy . a $\epsilon$ half-assed film . is $\epsilon$ clumsy , SL the writing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#States</th>
<th>Highest Scoring Phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>honest, and enjoyable soulful, scathing$<em>{SL}$ and joyous unpretentious, charming$</em>{SL}$, quirky forceful, and beautifully energetic, and surprisingly</td>
</tr>
<tr>
<td>3</td>
<td>five minutes four minutes final minutes first half-hour fifteen minutes</td>
</tr>
</tbody>
</table>

Back to [main]