Acquiring Lexical Semantic Knowledge

...And exploring ways to use it in applications

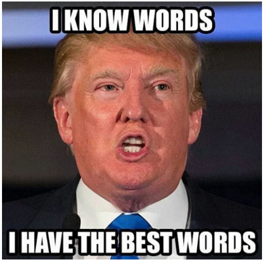
Vered Shwartz

Talk at Google Research IL, November 9, 2017

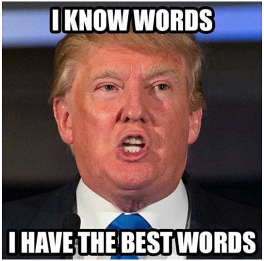
B I U
N L P

Outline

What is “lexical knowledge”?

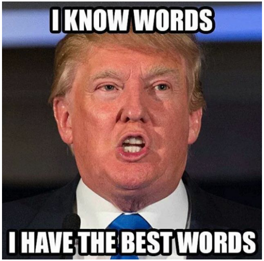


What is "lexical knowledge"?



- Knowledge about how words **relate** to each other.

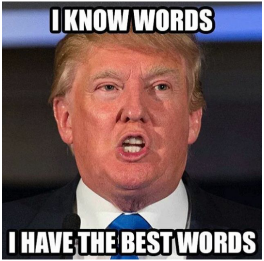
What is "lexical knowledge"?



- Knowledge about how words **relate** to each other.

- Valuable for making inferences:
  - *"pets are allowed"* $\Rightarrow$ *"dogs are allowed"*
  - *"dogs are allowed"* ?? *"pets are allowed"*

What is "lexical knowledge"?



- Knowledge about how words **relate** to each other.

- Valuable for making inferences:
  - *"pets are allowed"* $\Rightarrow$ *"dogs are allowed"*
  - *"dogs are allowed"* ?? *"pets are allowed"*
  - *"restaurant in Tel Aviv"* $\Rightarrow$ *"restaurant in Israel"*
  - *"restaurant in Israel"* ?? *"restaurant in Tel Aviv"*

Word Embeddings

First, let's get this off the table: ***"why not just use word embeddings?"***

Word Embeddings

First, let's get this off the table: ***"why not just use word embeddings?"***

- Word embeddings are great in capturing semantic relatedness!

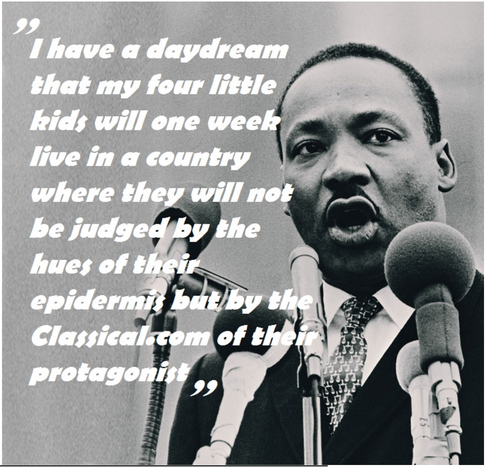
Word Embeddings

First, let's get this off the table: ***"why not just use word embeddings?"***

- Word embeddings are great in capturing semantic relatedness!
- ...but they mix all semantic relations together.

Word Embeddings

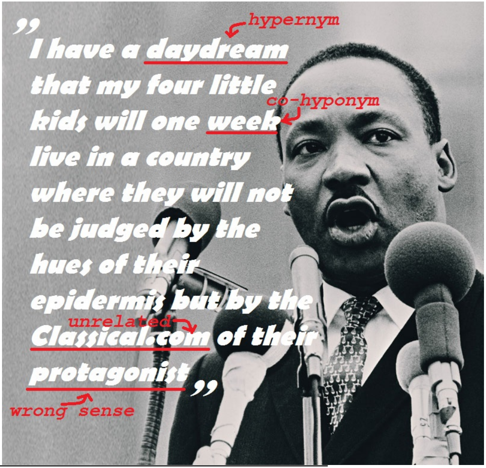
- To illustrate, take famous texts and replace nouns with their word2vec neighbours:[1]



[1] More examples here: `https://goo.gl/LJHzbi`

Word Embeddings

- To illustrate, take famous texts and replace nouns with their word2vec neighbours:[1]



[1] More examples here: https://goo.gl/LJHzbi

Acquiring Lexical Knowledge

Recognizing Semantic Relations between Nouns

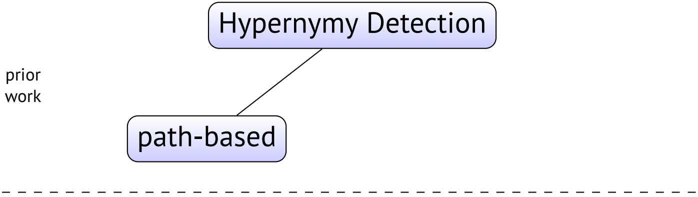
The Hypernymy Detection Task

- We first focused on hypernymy
  - The hyponym is a subclass of / instance of the hypernym
  - *(cat, animal)*, *(Google, company)*

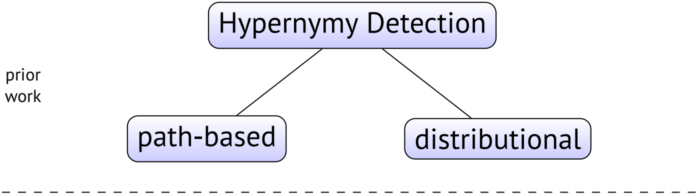
The Hypernymy Detection Task

- We first focused on hypernymy
  - The hyponym is a subclass of / instance of the hypernym
  - *(cat, animal)*, *(Google, company)*
- Given two terms, x and y, decide whether y is a hypernym of x
  - in some senses of x and y, e.g. *(apple, fruit), (apple, company)*

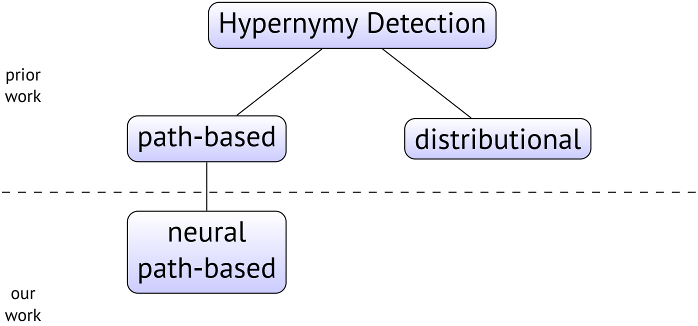
Corpus-based Hypernymy Detection



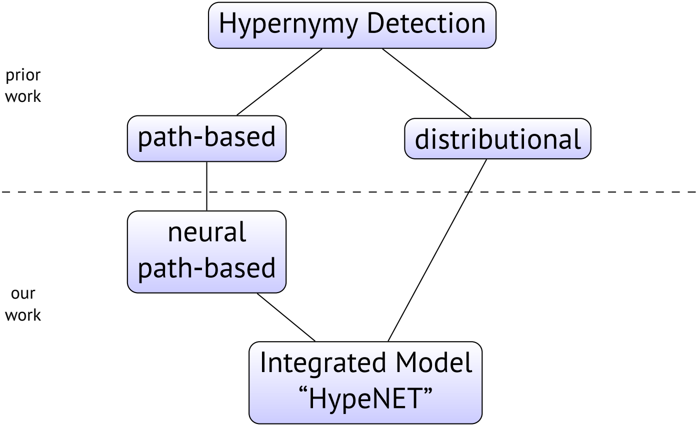
Corpus-based Hypernymy Detection



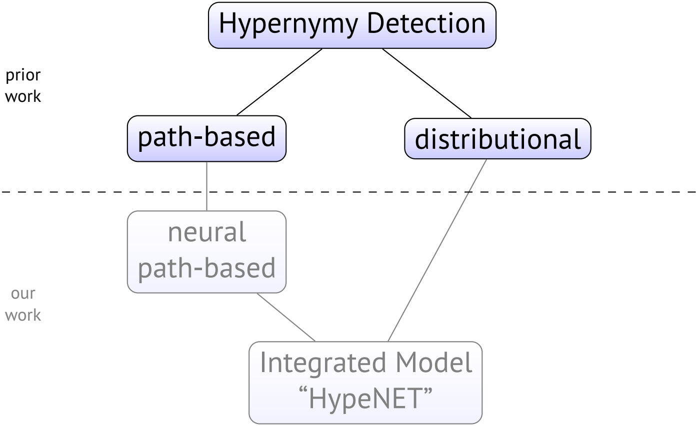
Corpus-based Hypernymy Detection



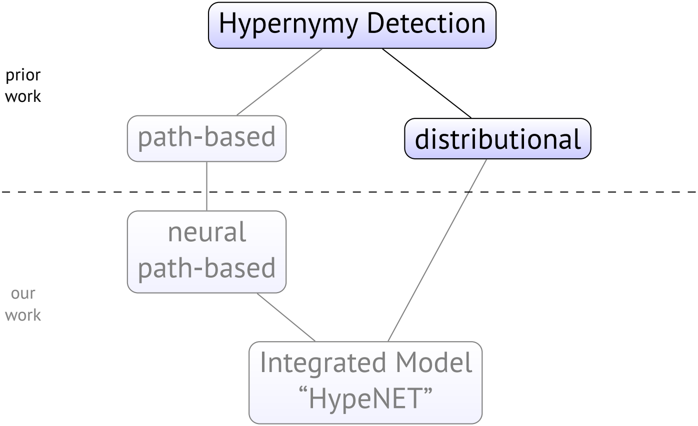
Corpus-based Hypernymy Detection



Prior Methods



Distributional Approach



Supervised Distributional Methods

- Recognize the relation between x and y based on their *separate* occurrences in the corpus

Supervised Distributional Methods

- Recognize the relation between x and y based on their *separate* occurrences in the corpus
- Represent (x, y) as a feature vector, based of the terms' embeddings:
  - Concatenation $\vec{x} \oplus \vec{y}$ [Baroni et al., 2012]
  - Difference $\vec{y} - \vec{x}$ [Roller et al., 2014, Weeds et al., 2014]

Supervised Distributional Methods

- Recognize the relation between x and y based on their *separate* occurrences in the corpus
- Represent (x, y) as a feature vector, based of the terms' embeddings:
  - Concatenation $\vec{x} \oplus \vec{y}$ [Baroni et al., 2012]
  - Difference $\vec{y} - \vec{x}$ [Roller et al., 2014, Weeds et al., 2014]
- Train a classifier to predict whether y is a hypernym of x

Supervised Distributional Methods

- Recognize the relation between x and y based on their *separate* occurrences in the corpus
- Represent (x, y) as a feature vector, based of the terms' embeddings:
  - Concatenation $\vec{x} \oplus \vec{y}$ [Baroni et al., 2012]
  - Difference $\vec{y} - \vec{x}$ [Roller et al., 2014, Weeds et al., 2014]
- Train a classifier to predict whether y is a hypernym of x

- Achieved very good results on common hypernymy detection datasets

Supervised Distributional Methods

- Recognize the relation between x and y based on their *separate* occurrences in the corpus
- Represent (x, y) as a feature vector, based of the terms' embeddings:
  - Concatenation $\vec{x} \oplus \vec{y}$ [Baroni et al., 2012]
  - Difference $\vec{y} - \vec{x}$ [Roller et al., 2014, Weeds et al., 2014]
- Train a classifier to predict whether y is a hypernym of x

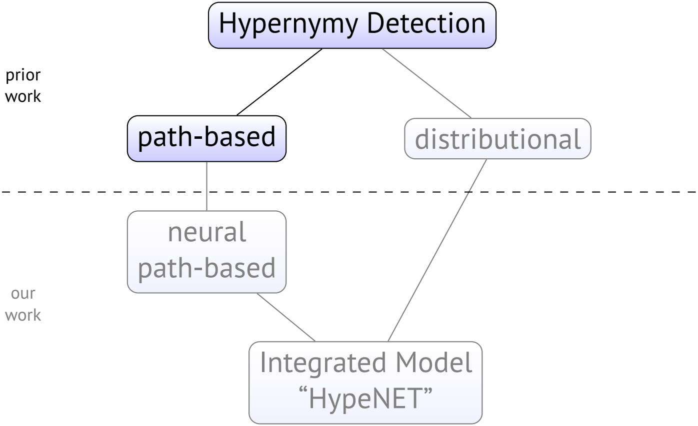
- Achieved very good results on common hypernymy detection datasets
- Is it a solved task?

Supervised Distributional Methods

- Recognize the relation between x and y based on their *separate* occurrences in the corpus
- Represent (x, y) as a feature vector, based of the terms' embeddings:
  - Concatenation $\vec{x} \oplus \vec{y}$ [Baroni et al., 2012]
  - Difference $\vec{y} - \vec{x}$ [Roller et al., 2014, Weeds et al., 2014]
- Train a classifier to predict whether y is a hypernym of x

- Achieved very good results on common hypernymy detection datasets
- Is it a solved task?
- Probably not. They don't learn the *relation* between x and y, but mostly that y is a *prototypical hypernym* [Levy et al., 2015].
  - e.g. that *(x, fruit)* or *(x, animal)* are always hypernyms

Path-based Approach



Path-based Approach

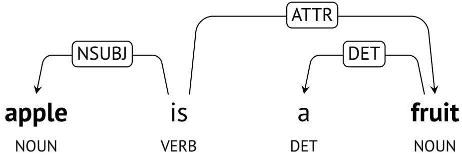
- Recognize the relation between *x* and *y* based on their *joint* occurrences in the corpus

Path-based Approach

- Recognize the relation between *x* and *y* based on their *joint* occurrences in the corpus
- Hearst Patterns [Hearst, 1992] - patterns connecting *x* and *y* may indicate that *y* is a hypernym of *x*
    - e.g. *X or other Y*, *X is a Y*, *Y, including X*

Path-based Approach

- Recognize the relation between *x* and *y* based on their *joint* occurrences in the corpus
- Hearst Patterns [Hearst, 1992] - patterns connecting *x* and *y* may indicate that *y* is a hypernym of *x*
  - e.g. *X or other Y*, *X is a Y*, *Y, including X*
- Patterns can be represented using dependency paths:



Supervised Path-based Approach

- Supervised method to recognize hypernymy [Snow et al., 2004]:

Supervised Path-based Approach

- Supervised method to recognize hypernymy [Snow et al., 2004]:
  - Features: all dependency paths that connected *x* and *y* in a corpus:

0	0	...	58	0	...	97	0	...	0
			↑ X and other Y			↑ such Y as X			

Supervised Path-based Approach

- Supervised method to recognize hypernymy [Snow et al., 2004]:
  - Features: all dependency paths that connected *x* and *y* in a corpus:

0	0	...	58	0	...	97	0	...	0

X and other Y   such Y as X

  - Trained a logistic regression classifier to predict hypernymy

Path-based Approach Issues

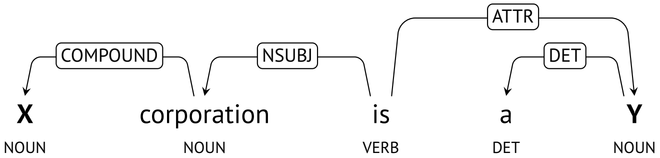
- The feature space is too sparse:

Path-based Approach Issues

- The feature space is too sparse:
  - Similar paths share no information:
    X inc. is a Y
    X group is a Y
    X organization is a Y

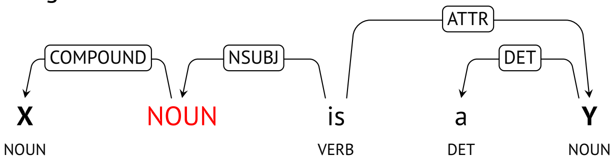
Path-based Approach Issues

- The feature space is too sparse:
  - Similar paths share no information:
    X inc. is a Y
    X group is a Y
    X organization is a Y
- PATTY [Nakashole et al., 2012] generalized paths, by replacing a word by:



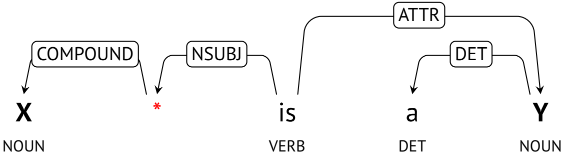
Path-based Approach Issues

- The feature space is too sparse:
  - Similar paths share no information:
    X inc. is a Y
    X group is a Y
    X organization is a Y
- PATTY [Nakashole et al., 2012] generalized paths, by replacing a word by:
  - its POS tag



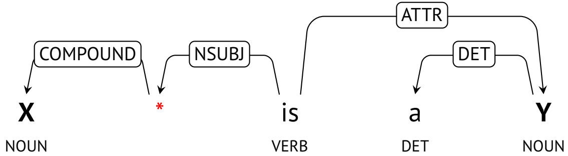
Path-based Approach Issues

- The feature space is too sparse:
  - Similar paths share no information:
    X inc. is a Y
    X group is a Y
    X organization is a Y
- PATTY [Nakashole et al., 2012] generalized paths, by replacing a word by:
  - a wild-card



Path-based Approach Issues

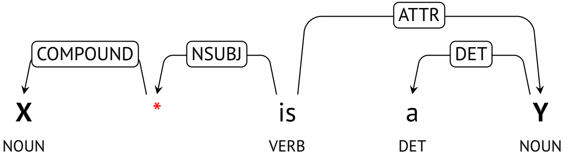
- The feature space is too sparse:
  - Similar paths share no information:
    X inc. is a Y
    X group is a Y
    X organization is a Y
- PATTY [Nakashole et al., 2012] generalized paths, by replacing a word by:
  - a wild-card



- Some of these generalizations are too general:
  - X is defined as Y ≈ X is described as Y via X is VERB as Y

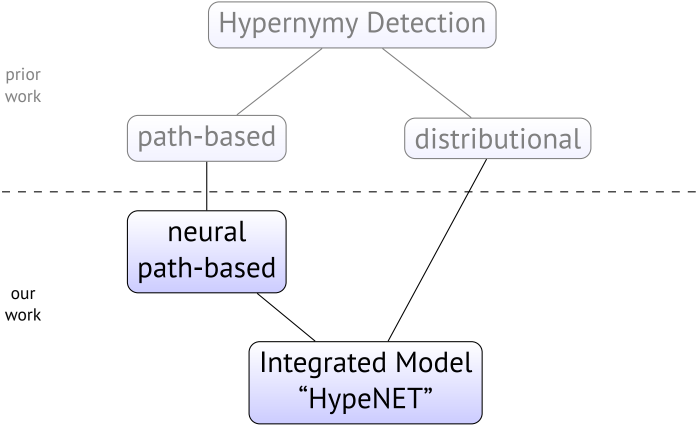
Path-based Approach Issues

- The feature space is too sparse:
  - Similar paths share no information:
    X inc. is a Y
    X group is a Y
    X organization is a Y
- PATTY [Nakashole et al., 2012] generalized paths, by replacing a word by:
  - a wild-card

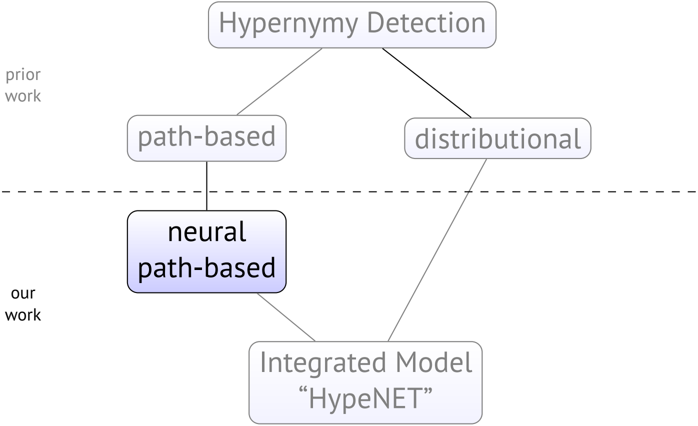


- Some of these generalizations are too general:
  - X is defined as Y ≈ X is described as Y via X is VERB as Y
  - X is defined as Y ≠ X is rejected as Y

HypeNET: Integrated Path-based and Distributional Method [Shwartz et al., 2016]



First Step: Improving Path Representation



Path Representation (1/2)

1. Split each path to edges

X	is	a	Y	⇒
‘X/NOUN/nsubj/>	be/VERB/ROOT/-		Y/NOUN/attr/<’	⇒
‘X/NOUN/nsubj/>’	‘be/VERB/ROOT/-’		‘Y/NOUN/attr/<’	

- Each edge consists of 4 components:

  dependent lemma / dependent POS / dependency label / direction

Path Representation (1/2)

1. Split each path to edges

```
       X                  is           a          Y          ⇒
    'X/NOUN/nsubj/> be/VERB/ROOT/- Y/NOUN/attr/<'            ⇒
'X/NOUN/nsubj/>'  'be/VERB/ROOT/-'     'Y/NOUN/attr/<'
```

- Each edge consists of 4 components:

  `dependent lemma` / `dependent POS` / `dependency label` / `direction`

- We learn embedding vectors for each component
  - Lemma embeddings are initialized with pre-trained word embeddings

Path Representation (1/2)

1. Split each path to edges

X	is	a	Y	⇒
‘X/NOUN/nsubj/>	be/VERB/ROOT/-	Y/NOUN/attr/<’		⇒
‘X/NOUN/nsubj/>’	‘be/VERB/ROOT/-’	‘Y/NOUN/attr/<’		

- Each edge consists of 4 components:

  `dependent lemma` / `dependent POS` / `dependency label` / `direction`

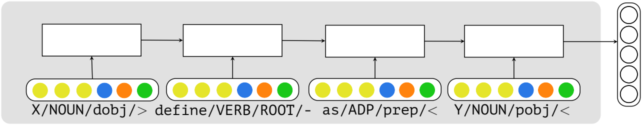
- We learn embedding vectors for each component
  - Lemma embeddings are initialized with pre-trained word embeddings
- The edge’s vector is the concatenation of its components’ vectors:

be/VERB/ROOT/-

  - Generalization: similar edges should have similar vectors!

Path Representation (2/2)

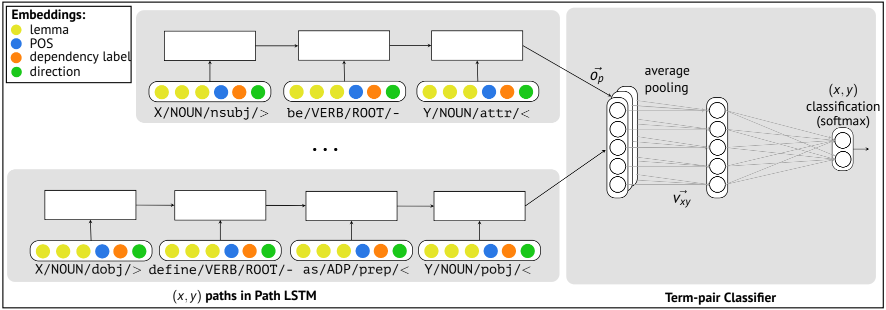
2. Feed the edges sequentially to an LSTM



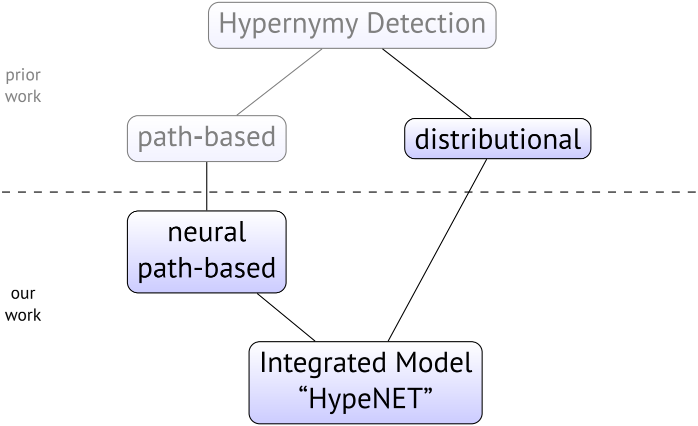
- Use the last output vector as the path embedding
- The LSTM may focus on edges that are more informative for the classification task, while ignoring others

Term-pair Classification

- The LSTM encodes a single path
- Each term-pair has multiple paths
  - Represent a term-pair as its averaged path embedding
- Classify for hypernymy (path-based network):



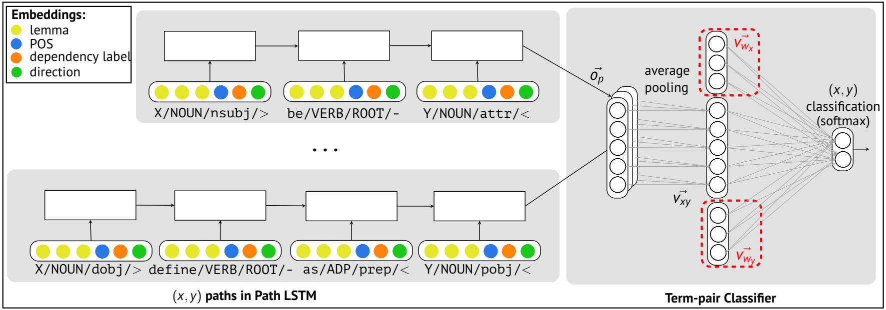
Second Step: Integrating Distributional Information



Second Step: Integrating Distributional Information

- Integrated network: add distributional information
  - Simply concatenate *x* and *y*'s word embeddings to the averaged path

- Classify for hypernymy (integrated network):



Results

- On a new dataset, built from knowledge resources

method		precision	recall	F_1
Path-based	Snow	0.843	0.452	0.589
	Snow + GEN	0.852	0.561	0.676
	HypeNET Path-based	0.811	0.716	0.761
Distributional	Best Supervised	0.901	0.637	0.746
Combined	HypeNET Integrated	**0.913**	**0.890**	**0.901**

- Path-based:
  - Compared to Snow + Snow with PATTY style generalizations
  - Our method outperforms path-based baselines with improved recall

Results

- On a new dataset, built from knowledge resources

method		precision	recall	F_1
Path-based	Snow	0.843	0.452	0.589
	Snow + GEN	0.852	0.561	0.676
	HypeNET Path-based	0.811	0.716	0.761
Distributional	Best Supervised	0.901	0.637	0.746
Combined	HypeNET Integrated	**0.913**	**0.890**	**0.901**

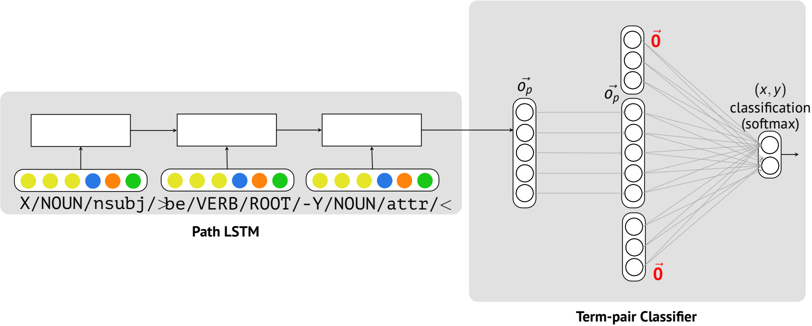
- The integrated method substantially outperforms both path-based and distributional methods

Analysis - Path Representation (1/2)

- Identify hypernymy-indicating paths:
  - Baselines: according to logistic regression feature weights

Analysis - Path Representation (1/2)

- Identify hypernymy-indicating paths:
  - Baselines: according to logistic regression feature weights
  - HypeNET: measure path contribution to positive classification:



- Take the top scoring paths according to $softmax(W \cdot [\vec{0}, \vec{o_p}, \vec{0}])[1]$

Analysis - Path Representation (2/2)

- Snow's method finds certain common paths:

  X company is a Y
  X ltd is a Y

Analysis - Path Representation (2/2)

- Snow's method finds certain common paths:

  X company is a Y
  X ltd is a Y

- PATTY-style generalizations find very general, possibly noisy paths:

  X NOUN is a Y

Analysis - Path Representation (2/2)

- Snow's method finds certain common paths:

  X company is a Y
  X ltd is a Y

- PATTY-style generalizations find very general, possibly noisy paths:

  X NOUN is a Y

- HypeNET makes fine-grained generalizations:

  X association is a Y
  X co. is a Y
  X company is a Y
  X corporation is a Y
  X foundation is a Y
  X group is a Y
  ...

Other Semantic Relations

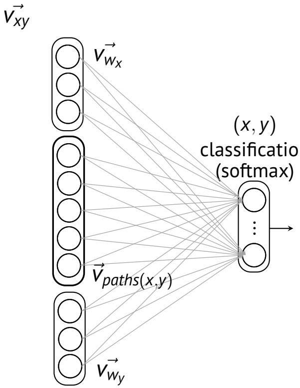
Recognizing Lexical Semantic Relations

- Given two terms, *x* and *y*, decide what is the semantic relation that holds between them (if any)
  - in some senses of *x* and *y*
  - e.g. both *fruit* and *company* are hypernyms of *apple*

LexNET - Multiple Semantic Relation Classification [Shwartz and Dagan, 2016a, Shwartz and Dagan, 2016b]

- Application of HypeNET for multiple relations:

  hypernymy, meroynymy, co-hyponymy, event, attribute, synonymy, antonymy, random



Results and Analysis

- LexNET outperforms individual path-based and distributional methods

Results and Analysis

- LexNET outperforms individual path-based and distributional methods
- Path-based contribution over distributional info is small when lexical memorization is enabled

Results and Analysis

- LexNET outperforms individual path-based and distributional methods
- Path-based contribution over distributional info is small when lexical memorization is enabled
- It is prominent in the following scenarios:
  - *x* or *y* are polysemous, e.g. *mero:(piano, key)*.

Results and Analysis

- LexNET outperforms individual path-based and distributional methods
- Path-based contribution over distributional info is small when lexical memorization is enabled
- It is prominent in the following scenarios:
  - *x* or *y* are polysemous, e.g. *mero:(piano, key)*.
  - the relation is not prototypical, e.g. *event:(cherry, pick)*.

Results and Analysis

- LexNET outperforms individual path-based and distributional methods
- Path-based contribution over distributional info is small when lexical memorization is enabled
- It is prominent in the following scenarios:
  - *x* or *y* are polysemous, e.g. *mero:(piano, key)*.
  - the relation is not prototypical, e.g. *event:(cherry, pick)*.
  - *x* or *y* are rare, e.g. *hyper:(mastodon, proboscidean)*.

Results and Analysis

- LexNET outperforms individual path-based and distributional methods
- Path-based contribution over distributional info is small when lexical memorization is enabled
- It is prominent in the following scenarios:
  - *x* or *y* are polysemous, e.g. *mero:(piano, key)*.
  - the relation is not prototypical, e.g. *event:(cherry, pick)*.
  - *x* or *y* are rare, e.g. *hyper:(mastodon, proboscidean)*.
- Thanks to the path representation, such relations are captured even with a single meaningful co-occurrence of *x* and *y*

Limitations

- All methods and baselines are bad in recognizing synonyms and antonyms.

Limitations

- All methods and baselines are bad in recognizing synonyms and antonyms.
  - **Path-based:**
    - Synonyms do not tend to occur together

Limitations

- All methods and baselines are bad in recognizing synonyms and antonyms.
  - **Path-based:**
    - Synonyms do not tend to occur together
    - Antonyms occur in similar paths as co-hyponyms:
      hot ***and*** *cold*, *cats* ***and*** *dogs*

Limitations

- All methods and baselines are bad in recognizing synonyms and antonyms.
  - **Path-based:**
    - Synonyms do not tend to occur together
    - Antonyms occur in similar paths as co-hyponyms:
      *hot **and** cold, cats **and** dogs*
  - **Distributional:**
    - Synonyms and antonyms occur in similar contexts:
      "go down in the elevator/lift", "it is hot/cold today"

Limitations

- All methods and baselines are bad in recognizing synonyms and antonyms.
  - **Path-based:**
    - Synonyms do not tend to occur together
    - Antonyms occur in similar paths as co-hyponyms:
      hot ***and*** *cold*, *cats* ***and*** *dogs*
  - **Distributional:**
    - Synonyms and antonyms occur in similar contexts:
      "go down in the elevator/lift", *"it is hot/cold today"*
- [Nguyen et al., 2017] used the method successfully to distinguish only between synonyms and antonyms.

Limitations

- All methods and baselines are bad in recognizing synonyms and antonyms.
  - **Path-based:**
    - Synonyms do not tend to occur together
    - Antonyms occur in similar paths as co-hyponyms: *hot* ***and*** *cold*, *cats* ***and*** *dogs*
  - **Distributional:**
    - Synonyms and antonyms occur in similar contexts: *"go down in the elevator/lift"*, *"it is hot/cold today"*
- [Nguyen et al., 2017] used the method successfully to distinguish only between synonyms and antonyms.
- [Rajana et al., 2017] integrated morphological cues (negated prefixes) to distinguish antonymy from other relations.

Interpreting Noun-Compounds

- Given a noun-compound $w_1 w_2$, classify the relation between the head w_2 and the modifier w_1
  - to one of a set of pre-defined relations
  - e.g. *olive oil* $\rightarrow$ `source`, *baby oil* $\rightarrow$ `purpose`

Interpreting Noun-Compounds

- Given a noun-compound $w_1 w_2$, classify the relation between the head w_2 and the modifier w_1
  - to one of a set of pre-defined relations
  - e.g. *olive oil* $\rightarrow$ `source`, *baby oil* $\rightarrow$ `purpose`

- Similar yet different from **semantic relation classification**:
  - We are interested in the relation between *olive* and *oil* in the context of the noun-compound, not in general

Interpreting Noun-Compounds

Previous Approaches

- **Paraphrasing:** Find joint corpus occurrences of w_1 and w_2, use paraphrases as features
  - e.g.: [w_2] *obtained from* [w_1] *(oil obtained from olives)*

Interpreting Noun-Compounds

Previous Approaches

- **Paraphrasing:** Find joint corpus occurrences of w_1 and w_2, use paraphrases as features
  - e.g.: [w_2] *obtained from* [w_1] *(oil obtained from olives)*
  - Problem: too sparse. e.g. [w_2] *extracted from* [w_1]

Interpreting Noun-Compounds

Previous Approaches

- **Paraphrasing:** Find joint corpus occurrences of w_1 and w_2, use paraphrases as features
  - e.g.: [w_2] *obtained from* [w_1] *(oil obtained from olives)*
  - Problem: too sparse. e.g. [w_2] *extracted from* [w_1]

- **Distributional:** Noun-compound representation as a function of w_1 and w_2 distributional representations

Interpreting Noun-Compounds

Previous Approaches

- **Paraphrasing:** Find joint corpus occurrences of w_1 and w_2, use paraphrases as features
  - e.g.: [w_2] *obtained from* [w_1] *(oil obtained from olives)*
  - Problem: too sparse. e.g. [w_2] *extracted from* [w_1]

- **Distributional:** Noun-compound representation as a function of w_1 and w_2 distributional representations
  - Problem: memorizes common relations of w_1 and w_2 separately

(**lexical memorization**)

Interpreting Noun-Compounds
[Shwartz and Waterson, in preparation]

- We applied LexNET to this task

Interpreting Noun-Compounds
[Shwartz and Waterson, in preparation]

- We applied LexNET to this task
- LexNET improves performance:
  - On a lexical split dataset (i.e. not allowing lexical memorization)
  - On a new, challenging dataset we created

Interpreting Noun-Compounds
[Shwartz and Waterson, in preparation]

- We applied LexNET to this task
- LexNET improves performance:
  - On a lexical split dataset (i.e. not allowing lexical memorization)
  - On a new, challenging dataset we created
- Performs worse than the baseline when lexical memorization is possible

Interpreting Noun-Compounds
[Shwartz and Waterson, in preparation]

- We applied LexNET to this task
- LexNET improves performance:
  - On a lexical split dataset (i.e. not allowing lexical memorization)
  - On a new, challenging dataset we created
- Performs worse than the baseline when lexical memorization is possible
- In general, the task is very difficult:
  - Lots of relations
  - Some relations have no indicative paths (e.g. `non-compositional`)

Acquiring Predicate Paraphrases

Acquiring Predicate Paraphrases from News Tweets [Shwartz et al., 2017][2]

$[a]_0$ introduce $[a]_1$	$[a]_0$ welcome $[a]_1$
$[a]_0$ appoint $[a]_1$	$[a]_0$ to become $[a]_1$
$[a]_0$ die at $[a]_1$	$[a]_0$ pass away at $[a]_1$
$[a]_0$ hit $[a]_1$	$[a]_0$ sink to $[a]_1$
$[a]_0$ be investigate $[a]_1$	$[a]_0$ be probe $[a]_1$
$[a]_0$ eliminate $[a]_1$	$[a]_0$ slash $[a]_1$
$[a]_0$ announce $[a]_1$	$[a]_0$ unveil $[a]_1$
$[a]_0$ quit after $[a]_1$	$[a]_0$ resign after $[a]_1$
$[a]_0$ announce as $[a]_1$	$[a]_0$ to become $[a]_1$
$[a]_0$ threaten $[a]_1$	$[a]_0$ warn $[a]_1$
$[a]_0$ die at $[a]_1$	$[a]_0$ live until $[a]_1$
$[a]_0$ double down on $[a]_1$	$[a]_0$ stand by $[a]_1$
$[a]_0$ kill $[a]_1$	$[a]_0$ shoot $[a]_1$
$[a]_0$ approve $[a]_1$	$[a]_0$ pass $[a]_1$
seize $[a]_0$ at $[a]_1$	to grab $[a]_0$ at $[a]_1$

- Binary verbal predicate paraphrases

[2] Available at `https://github.com/vered1986/Chirps`

Acquiring Predicate Paraphrases from News Tweets [Shwartz et al., 2017][2]

$[a]_0$ introduce $[a]_1$	$[a]_0$ welcome $[a]_1$
$[a]_0$ appoint $[a]_1$	$[a]_0$ to become $[a]_1$
$[a]_0$ die at $[a]_1$	$[a]_0$ pass away at $[a]_1$
$[a]_0$ hit $[a]_1$	$[a]_0$ sink to $[a]_1$
$[a]_0$ be investigate $[a]_1$	$[a]_0$ be probe $[a]_1$
$[a]_0$ eliminate $[a]_1$	$[a]_0$ slash $[a]_1$
$[a]_0$ announce $[a]_1$	$[a]_0$ unveil $[a]_1$
$[a]_0$ quit after $[a]_1$	$[a]_0$ resign after $[a]_1$
$[a]_0$ announce as $[a]_1$	$[a]_0$ to become $[a]_1$
$[a]_0$ threaten $[a]_1$	$[a]_0$ warn $[a]_1$
$[a]_0$ die at $[a]_1$	$[a]_0$ live until $[a]_1$
$[a]_0$ double down on $[a]_1$	$[a]_0$ stand by $[a]_1$
$[a]_0$ kill $[a]_1$	$[a]_0$ shoot $[a]_1$
$[a]_0$ approve $[a]_1$	$[a]_0$ pass $[a]_1$
seize $[a]_0$ at $[a]_1$	to grab $[a]_0$ at $[a]_1$

- Binary verbal predicate paraphrases
- Extracted from Twitter

[2] Available at `https://github.com/vered1986/Chirps`

Acquiring Predicate Paraphrases from News Tweets [Shwartz et al., 2017][2]

$[a]_0$ introduce $[a]_1$	$[a]_0$ welcome $[a]_1$
$[a]_0$ appoint $[a]_1$	$[a]_0$ to become $[a]_1$
$[a]_0$ die at $[a]_1$	$[a]_0$ pass away at $[a]_1$
$[a]_0$ hit $[a]_1$	$[a]_0$ sink to $[a]_1$
$[a]_0$ be investigate $[a]_1$	$[a]_0$ be probe $[a]_1$
$[a]_0$ eliminate $[a]_1$	$[a]_0$ slash $[a]_1$
$[a]_0$ announce $[a]_1$	$[a]_0$ unveil $[a]_1$
$[a]_0$ quit after $[a]_1$	$[a]_0$ resign after $[a]_1$
$[a]_0$ announce as $[a]_1$	$[a]_0$ to become $[a]_1$
$[a]_0$ threaten $[a]_1$	$[a]_0$ warn $[a]_1$
$[a]_0$ die at $[a]_1$	$[a]_0$ live until $[a]_1$
$[a]_0$ double down on $[a]_1$	$[a]_0$ stand by $[a]_1$
$[a]_0$ kill $[a]_1$	$[a]_0$ shoot $[a]_1$
$[a]_0$ approve $[a]_1$	$[a]_0$ pass $[a]_1$
seize $[a]_0$ at $[a]_1$	to grab $[a]_0$ at $[a]_1$

- Binary verbal predicate paraphrases
- Extracted from Twitter
- Ever-growing resource: currently around 1.5M paraphrases

[2] Available at `https://github.com/vered1986/Chirps`

Assumptions

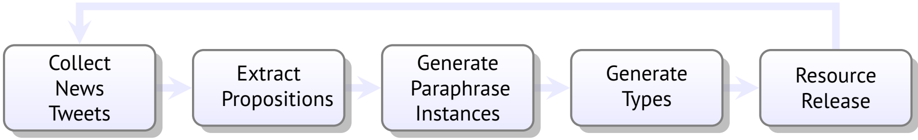
- **Main assumption:** redundant news headlines of the same event are likely to describe it with different words [Shinyama et al., 2002, Barzilay and Lee, 2003].

Assumptions

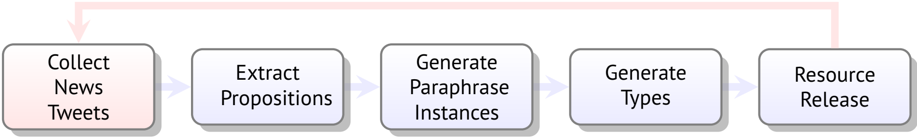
- **Main assumption:** redundant news headlines of the same event are likely to describe it with different words [Shinyama et al., 2002, Barzilay and Lee, 2003].
- **This work:** propositions extracted from tweets discussing news events, published on the same day, that agree on their arguments, are predicate paraphrases.

[Amazon] to buy / is buying / to acquire [Whole Foods]

Resource Collection



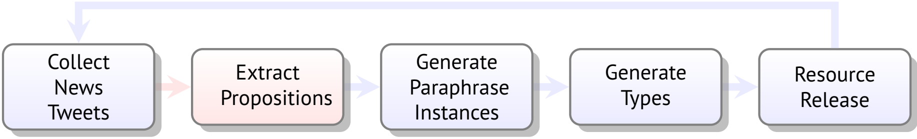
Resource Collection



- Query the Twitter Search API for news tweets in English

Amazon is buying Whole Foods in $13.7B
Amazon to acquire Whole Foods Market in deal valued at nearly $14 billion
. . .

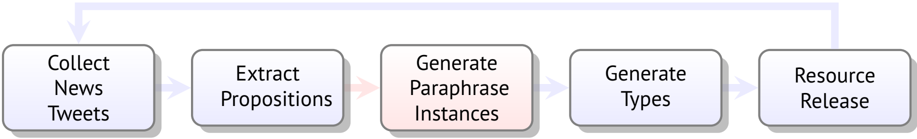
Resource Collection



- Extract propositions from tweets using PropS [Stanovsky et al., 2016]
- Get binary verbal predicate templates, and apply argument reduction [Stanovsky and Dagan, 2016]

[Amazon] **buy** [Whole Foods]
[Amazon] **acquire** [Whole Foods Market]
...

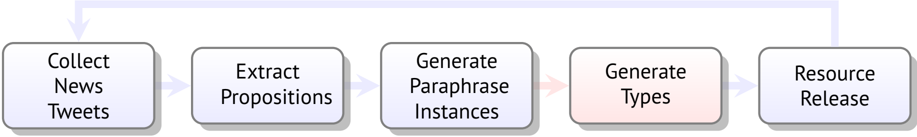
Resource Collection



- We consider two predicates as paraphrases if:
  1. They appear on the same day.
  2. Each of their arguments aligns with a unique argument in the other predicate.
- Two levels of argument matching: **strict** (exact match / short edit distance) and **loose** (partial token matching / WordNet synonyms)

$[a]_0$ **buy** $[a]_1$	$[a]_0$ **acquire** $[a]_1$	Amazon	Whole Foods
$[a]_0$ **buy** $[a]_1$	$[a]_0$ **acquire** $[a]_1$	Intel	Mobileye
	...		

Resource Collection



Heuristic score for a predicate paraphrase type:

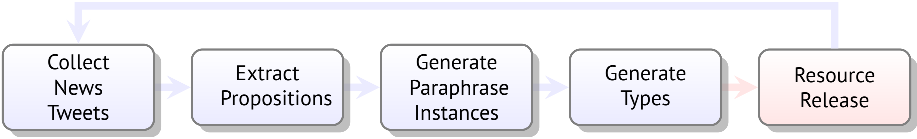
$$p_1 = [a]_0 \textbf{ buy } [a]_1, \quad p_2 = [a]_0 \textbf{ acquire } [a]_1$$

$$s(p_1, p_2) = count(p_1, p_2) \cdot \left(1 + \frac{days(p_1, p_2)}{N}\right)$$

- $count(p_1, p_2)$ assigns high scores for frequent paraphrases
- N - number of days since the resource collection begun
- $\frac{days(p_1, p_2)}{N}$ eliminates noise from two arguments participating in different events on the same day

  1) *Last year when Chuck Berry turned 90*; 2) *Chuck Berry dies at 90*

Resource Collection



- We release our resource daily, with two files:
  - **Instances**: predicates, arguments and tweet IDs.
  - **Types**: predicate paraphrase pair types ranked in a descending order according to the heuristic accuracy score.

Using Lexical Knowledge in Sentence-level Applications

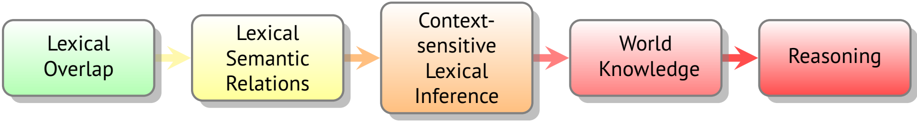
Sentence-level Inference

- RTE: given a premise *p* and a hypothesis *h*, can a reader reading *p* infer that *h* is likely true? [Dagan et al., 2013].
  - Very small datasets, unsuitable for today's neural models

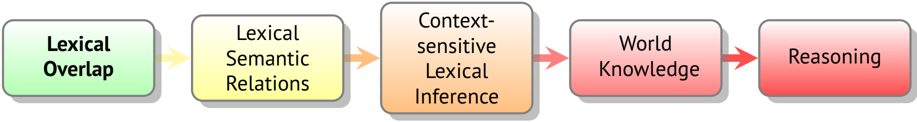
Sentence-level Inference

- RTE: given a premise *p* and a hypothesis *h*, can a reader reading *p* infer that *h* is likely true? [Dagan et al., 2013].
  - Very small datasets, unsuitable for today's neural models
- NLI: natural language inference - 3-way classification for entailment, neutral, and contradiction:
  - SNLI [Bowman et al., 2015]
  - MultiNLI [Williams et al., 2017]

Knowledge Required for Sentence-level Inference



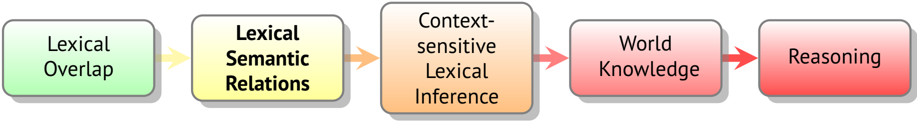
Knowledge Required for Sentence-level Inference



- Premise:
  Three young women embrace while displaying baked goods in kitchen.

- Hypothesis:
  Three young women embrace while they show off their baked goods to potential buyers.

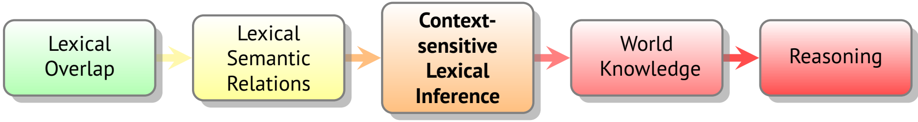
Knowledge Required for Sentence-level Inference



- Premise:
  Three young women embrace while displaying baked goods in kitchen.

- Hypothesis:
  Three young women embrace while they show off their baked goods to potential buyers.

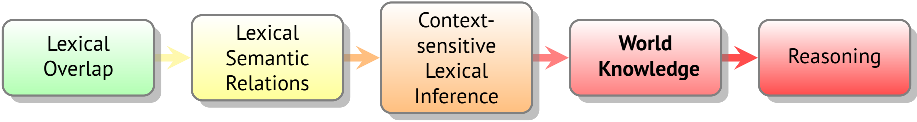
Knowledge Required for Sentence-level Inference



- Premise:
  Elderly bald man with a beard playing the guitar in a band.

- Hypothesis:
  There are people making music together.

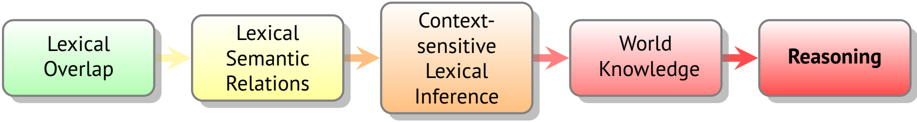
Knowledge Required for Sentence-level Inference



- Premise:
  A performer standing on a platform in Times Square.

- Hypothesis:
  The performer is in New York.

Knowledge Required for Sentence-level Inference



- Premise:
  In a train station, an attractive woman in a blue skirt and jacket, surrounded by her luggage, passes time with a crossword.

- Hypothesis:
  A woman is doing a crossword puzzle while waiting for a train.

Existing Solutions

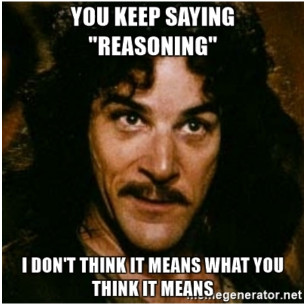
- Recent neural models are good with lexical overlap and reasonable with semantic relations.

Existing Solutions

- Recent neural models are good with lexical overlap and reasonable with semantic relations.
- Many papers claim to solve “reasoning”, but their success stems from the dataset being too easy.
  - e.g. high correlation between lexical overlap and entailment

Existing Solutions

- Recent neural models are good with lexical overlap and reasonable with semantic relations.
- Many papers claim to solve “reasoning”, but their success stems from the dataset being too easy.
  - e.g. high correlation between lexical overlap and entailment



Our Vision

- Goal: improve sentence-level inference with lexical knowledge

Our Vision

- Goal: improve sentence-level inference with lexical knowledge
- Means: inject knowledge into neural models to combine the best of both worlds

Our Vision

P: *An elderly man is drinking orange juice at a cafe.*
H: *An old man is sipping a beverage.*

1. Extract propositions:

Premise

[man] drink [orange juice]

[man] be at [cafe]

[man] be [elderly]

Hypothesis

[man] sip [beverage]

[man] be [old]

Our Vision

P: *An elderly man is drinking orange juice at a cafe.*
H: *An old man is sipping a beverage.*

2. Align arguments based on lexical semantic relations:

Premise

[man]$_1$ drink [orange juice]$_2$

[man]$_1$ be at [cafe]$_4$

[man]$_1$ be [elderly]$_3$

Hypothesis

[man]$_1$ sip [beverage]$_2$

[man]$_1$ be [old]$_3$

Our Vision

P: *An elderly man is drinking orange juice at a cafe.*
H: *An old man is sipping a beverage.*

3. Align propositions based on argument and predicate entailment:

Premise

[man]$_1$ drink [orange juice]$_2$

[man]$_1$ be at [cafe]$_4$

[man]$_1$ be [elderly]$_3$

Hypothesis

[man]$_1$ sip [beverage]$_2$

[man]$_1$ be [old]$_3$

Our Vision

P: *An elderly man is drinking orange juice at a cafe.*
H: *An old man is sipping a beverage.*

4. Make a sentence-level decision based on proposition alignment:

Premise

[man]$_1$ drink [orange juice]$_2$

[man]$_1$ be at [cafe]$_4$

[man]$_1$ be [elderly]$_3$

Entailment

Hypothesis

[man]$_1$ sip [beverage]$_2$

[man]$_1$ be [old]$_3$

Limitations and Drawbacks

- Difficult to show improvement on existing datasets
  - Current SOTA: SNLI - 90% accuracy, MultiNLI - 80% accuracy
  - Most models work on surface level, no external knowledge

Limitations and Drawbacks

- Difficult to show improvement on existing datasets
  - Current SOTA: SNLI - 90% accuracy, MultiNLI - 80% accuracy
  - Most models work on surface level, no external knowledge
- Tools and knowledge introduce new errors:
  - Parsing
  - Proposition extraction
  - Automatically-extracted lexical knowledge

Thank You!

References I

[Baroni et al., 2012] Baroni, M., Bernardi, R., Do, N.-Q., and Shan, C.-c. (2012). Entailment above the word level in distributional semantics. In *EACL*, pages 23–32.

[Barzilay and Lee, 2003] Barzilay, R. and Lee, L. (2003). Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *NAACL*.

[Bowman et al., 2015] Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

[Dagan et al., 2013] Dagan, I., Roth, D., Sammons, M., and Zanzotto, F. M. (2013). Recognizing textual entailment: Models and applications. *Synthesis Lectures on Human Language Technologies*, 6(4):1–220.

[Hearst, 1992] Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *ACL*, pages 539–545.

[Levy et al., 2015] Levy, O., Remus, S., Biemann, C., and Dagan, I. (2015). Do supervised distributional methods really learn lexical inference relations. *NAACL*.

[Nakashole et al., 2012] Nakashole, N., Weikum, G., and Suchanek, F. (2012). Patty: a taxonomy of relational patterns with semantic types. In *EMNLP and CoNLL*, pages 1135–1145.

[Nguyen et al., 2017] Nguyen, K. A., Schulte im Walde, S., and Vu, N. T. (2017). Distinguishing antonyms and synonyms in a pattern-based neural network. In *EACL*.

[Rajana et al., 2017] Rajana, S., Callison-Burch, C., Apidianaki, M., and Shwartz, V. (2017). Learning antonyms with paraphrases and a morphology-aware neural network. In **SEM*, pages 12–21.

[Roller et al., 2014] Roller, S., Erk, K., and Boleda, G. (2014). Inclusive yet selective: Supervised distributional hypernymy detection. In *COLING*, pages 1025–1036.

[Shinyama et al., 2002] Shinyama, Y., Sekine, S., and Sudo, K. (2002). Automatic paraphrase acquisition from news articles. In *HLT*, pages 313–318. Morgan Kaufmann Publishers Inc.

References II

[Shwartz and Dagan, 2016a] Shwartz, V. and Dagan, I. (2016a). path-based vs. distributional information in recognizing lexical semantic relations. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex-V), in COLING*, Osaka, Japan.

[Shwartz and Dagan, 2016b] Shwartz, V. and Dagan, I. (2016b). cogalex-v shared task: Lexnet - integrated path-based and distributional method for the identification of semantic relations. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex-V), in COLING*, Osaka, Japan.

[Shwartz et al., 2016] Shwartz, V., Goldberg, Y., and Dagan, I. (2016). Improving hypernymy detection with an integrated path-based and distributional method. In *ACL*, pages 2389–2398.

[Shwartz et al., 2017] Shwartz, V., Stanovsky, G., and Dagan, I. (2017). Acquiring predicate paraphrases from news tweets. In **SEM*, pages 155–160.

[Snow et al., 2004] Snow, R., Jurafsky, D., and Ng, A. Y. (2004). Learning syntactic patterns for automatic hypernym discovery. In *NIPS*.

[Stanovsky and Dagan, 2016] Stanovsky, G. and Dagan, I. (2016). Annotating and predicting non-restrictive noun phrase modifications. In *ACL*.

[Stanovsky et al., 2016] Stanovsky, G., Ficler, J., Dagan, I., and Goldberg, Y. (2016). Getting more out of syntax with props. *arXiv*.

[Weeds et al., 2014] Weeds, J., Clarke, D., Reffin, J., Weir, D., and Keller, B. (2014). Learning to distinguish hypernyms and co-hyponyms. In *COLING*, pages 2249–2259.

[Williams et al., 2017] Williams, A., Nangia, N., and Bowman, S. R. (2017). A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.