Triple-based Background Knowledge Ranking for Document Enrichment

Muyu Zhang, Bing Qin, Ting Liu, and Mao Zheng

HARBIN INSTITUTE OF TECHNOLOGY
Our Task

To find and rank relevant background knowledge in the form of triple

**Input**: one source document and a large set of background Knowledge in the form of triple

**Output**: Top N relevant background knowledge
$S_1$: Coalition may never know if Iraqi president Saddam Hussein survived a U.S. air strike yesterday.
$S_1$: Coalition may never know if Iraqi president Saddam Hussein survived a U.S. air strike yesterday.

$S_2$: A B-1 bomber dropped four 2,000-pound bombs on a building in a residential area of Baghdad.
Source Document

$S_1$: Coalition may never know if Iraqi president Saddam Hussein survived a U.S. air strike yesterday.

$S_2$: A B-1 bomber dropped four 2,000-pound bombs on a building in a residential area of Baghdad.

$S_3$: They had got an intelligence reports senior officials were meeting there, possibly including Saddam Hussein and his sons.
**S₁:** Coalition may never know if Iraqi president Saddam Hussein survived a U.S. air strike yesterday.

**S₂:** A B-1 bomber dropped four 2,000-pound bombs on a building in a residential area of Baghdad.

**S₃:** They had got an intelligence reports senior officials were meeting there, possibly including Saddam Hussein and his sons.
The key is:

Background Knowledge!

But, these knowledge is available for human

NOT FOR COMPUTERS!
Background Knowledge:

“Saddam, liveIn, Baghdad”

“Iraqi, hasCapital, Baghdad”

“Saddam, hasChild, Qusay”

......

Source Document

$S_1$: Coalition may never know if Iraqi president Saddam Hussein survived a U.S. air strike yesterday.

$S_2$: A B-1 bomber dropped four 2,000-pound bombs on a building in a residential area of Baghdad.

$S_3$: They had got an intelligence reports senior officials were meeting there, possibly including Saddam Hussein and his sons.
**S_1**: Coalition may never know if Iraqi president Saddam Hussein survived a U.S. air strike yesterday.

**S_2**: A B-1 bomber dropped four 2,000-pound bombs on a building in a residential area of Baghdad.

**S_3**: They had got an intelligence reports senior officials were meeting there, possibly including Saddam Hussein and his sons.
S1: Coalition may never know if Iraqi president Saddam Hussein survived a U.S. air strike yesterday.

S2: A B-1 bomber dropped four 2,000-pound bombs on a building in a residential area of Baghdad.

S3: They had got an intelligence reports senior officials were meeting there, possibly including Saddam Hussein and his sons.
Why triple?

- We use background knowledge in the form of triple: “argument₁, predicate, argument₂”

- So we focus on finding and ranking on these triples

*Less noise and less ambiguity*
OUR SOLUTION ...
Questions

• Where knowledge comes from
• How to rank these knowledge
• Existing knowledge bases
  – YAGO (Hoffart et al., 2013)
  – 447,000,000 facts formed as “argument$_1$, predicate, argument$_2$” and partly manually edited
  – E.g. “Iraqi, hasCapital, Baghdad”
Where?

• Existing knowledge bases
  – YAGO (Hoffart et al., 2013)
  – 447,000,000 facts formed as “argument$_1$, predicate, argument$_2$” and partly manually edited
  – E.g. “Iraqi, hasCapital, Baghdad”

• Automatically extracted knowledge
  – Reverb (Etzioni et al., 2011)
  – Take raw text as input and automatically extract knowledge formed as “argument$_1$, predicate, argument$_2$”
  – E.g. “Saddam, return to live in, Baghdad”
Where?

- There is too much knowledge in the world
  - 447,000,000 facts in YAGO
  - Infinite knowledge generated during automatic extraction process
Where?

- There is too much knowledge in the world
  - 447,000,000 facts in YAGO
  - Infinite knowledge generated during automatic extraction process

We cannot treat all of them as candidates to be ranked. So, which to choose?
Where?

• There is too much knowledge in the world
  – 447,000,000 facts in YAGO
  – Infinite knowledge generated during automatic extraction process

  We cannot treat all of them as candidates to be ranked. So, which to choose?

• YAGO: lexically matched facts

2014/8/26 Tuesday
There is too much knowledge in the world

- 447,000,000 facts in YAGO
- Infinite knowledge generated during automatic extraction process

We cannot treat all of them as candidates to be ranked. So, which to choose?

- YAGO: lexically matched facts
- Automatic extraction: knowledge extracted from relevant documents
Questions

• Where knowledge comes from
• How to rank these knowledge
Basic Idea

- Source document consists of multiple information, which can be extracted as triples

$(document)$ ➔ $(sd-nodes)$

**$sd$-node**: source document information

**$bk$-node**: background knowledge
Basic Idea

• For certain background knowledge in the form of triple (bk-node), the relevance to source document is converted into relevance to its sd-nodes.
Basic Idea

- We present *sd-nodes* and *bk-nodes* together, then propagate relevance score from sd-nodes to bk-nodes
• We present \textit{sd-nodes} and \textit{bk-nodes} together, then propagate relevance score from sd-nodes to bk-nodes

\begin{itemize}
\item \textit{sd-nodes}
\item \textit{bk-nodes}
\end{itemize}
Basic Idea

- Given source document and certain `bk-node`, what decides their relevance?
Basic Idea

- Given source document and certain \textit{bk-node}, what decides their relevance?
  - F1: How many \textit{sd-nodes} are relevant to the \textit{bk-node}?
    - More sd-nodes -> more relevant
Basic Idea

• Given source document and certain **bk-node**, what decides their relevance?
  – F1: How many **sd-nodes** are relevant to the **bk-node**?
    • More sd-nodes -> more relevant
  – F2: How relevant is the **bk-node** to these **sd-nodes**?
    • More relevant to sd-nodes -> more relevant
Given source document and certain **bk-node**, what decides their relevance?

- **F1**: How many **sd-nodes** are relevant to the **bk-node**?
  - More sd-nodes -> more relevant

- **F2**: How relevant is the **bk-node** to these **sd-nodes**?
  - More relevant to sd-nodes -> more relevant

- **F3**: How important are these **sd-nodes**?
  - More important -> more relevant
How to rank?

– F1: How many *sd-nodes* are relevant to the *bk-node*?
  • More sd-nodes -> more relevant
How to rank?

– F1: How many sd-nodes are relevant to the bk-node?
  • More sd-nodes -> more relevant

– Solution
  • We add edges between bk-node and relevant sd-nodes
How to rank?

– F1: How many *sd-nodes* are relevant to the *bk-node*?
  • More sd-nodes -> more relevant

– Solution
  • We add edges between *bk-node* and relevant *sd-nodes*
How to rank?

– F2: How relevant is the *bk-node* to these *sd-nodes*?
  • More relevant to sd-nodes -> more relevant
How to rank?

– F2: How relevant is the **bk-node** to these **sd-nodes**?
  • More relevant to sd-nodes -> more relevant

– Solution
  • We evaluate the relevance between the **bk-node** and every **sd-node** with search engine
How to rank?

– F2: How relevant is the \textit{bk-node} to these \textit{sd-nodes}?  
  • More relevant to sd-nodes -> more relevant

– Solution
  • We evaluate the relevance between the \textit{bk-node} and every \textit{sd-node} with search engine

\[
WebJaccard(p, q) = \begin{cases} 
0 & \text{if } H(p \cap q) \leq C \\
\frac{H(p \cap q)}{H(p) + H(q) - H(p \cap q)} & \text{otherwise}.
\end{cases}
\]

\(H(P)\) indicates the number of pages returned by search engine, given the query \(P\).
How to rank?

– F2: How relevant is the *bk-node* to these *sd-nodes*?
  • More relevant to *sd-nodes* -> more relevant

– Solution
  • We evaluate the relevance between the *bk-node* and every *sd-node* with search engine

![Diagram showing relevance between nodes](image-url)
– F3: How important are these *sd-nodes*?
  • More important -> more relevant
How to rank?

– F3: How important are these \textit{sd-nodes}?
  • More important -> more relevant

– Solution
  • We evaluate the importance of \textit{sd-nodes} and assign higher initial value to important ones
How to rank?

– F3: How important are these *sd-nodes*?
  - More important -> more relevant

– Solution
  - We evaluate the importance of *sd-nodes* and assign higher initial value to important ones
How to rank?

Combine them together...
How to rank?

- Iterative relevance propagation over the graph
  
  \[ \vec{W}' = \vec{W} \times P \]

  \[ = \vec{W} \times \begin{bmatrix}
  p(1, 1) & p(1, 2) & \cdots & p(1, n) \\
  p(2, 1) & p(2, 2) & \cdots & p(2, n) \\
  \vdots & \vdots & \ddots & \vdots \\
  p(n, 1) & p(n, 2) & \cdots & p(n, n)
\end{bmatrix} \]

2014/8/26 Tuesday
How to rank?

- Iterative relevance propagation over the graph
  - Iterative propagation

\[
\tilde{W}' = \tilde{W} \times P
\]

\[
= \tilde{W} \times \begin{bmatrix}
  p(1, 1) & p(1, 2) & \cdots & p(1, n) \\
  p(2, 1) & p(2, 2) & \cdots & p(2, n) \\
  \cdots & \cdots & \cdots & \cdots \\
  p(n, 1) & p(n, 2) & \cdots & p(n, n)
\end{bmatrix}
\]

for every element

\[
w_i' = w_1 \cdot p(1, i) + w_2 \cdot p(2, i) + \cdots + w_n \cdot p(n, i)
\]

\[
= \sum_{k \in N} w_k \cdot p(k, i)
\]

\[
= \sum_{k \in N} w_k \cdot \left( \frac{r(i, j) \times \delta(i, j)}{\sum_{k \in N} r(k, j) \times \delta(k, j)} \right)
\]
How to rank?

• Iterative relevance propagation over the graph
  – Propagation probability

\[
p(i, j) = \frac{r(i, j) \times \delta(i, j)}{\sum_{k \in N} r(k, j) \times \delta(k, j)}
\]

where

\[
\delta(i, j) = \begin{cases} 
1 & \text{if } (i, j) \in E \\
0 & \text{otherwise}
\end{cases}
\]
How to rank?

• Iterative relevance propagation over the graph
  – Stop when a global stage is achieved
  – Rank all the background knowledge according to their relevance scores
  – Output the ranked list of background knowledge
EXPERIMENTS
Experiments

• Data Preparation
  – Source document: ACE corpus (Doddington et al., 2004)
  – Background Knowledge: YAGO + REVERB
Experiments

• Data Preparation
  – Source document: ACE corpus (Doddington et al., 2004)
  – Background Knowledge: YAGO + REVERB

• Ranked knowledge are checked by three annotators
Experiments

• Data Preparation
  – Source document: ACE corpus (Doddington et al., 2004)
  – Background Knowledge: YAGO + REVERB

• Ranked knowledge are checked by three annotators
  – Totally annotated 7,000 background knowledge
Experiments

• Data Preparation
  – Source document: ACE corpus (Doddington et al., 2004)
  – Background Knowledge: YAGO + REVERB

• Ranked knowledge are checked by three annotators
  – Totally annotated 7,000 background knowledge
  – When confliction happens, we choose the label selected by more annotators
Experiments

• Data Preparation
  – Source document: ACE corpus (Doddington et al., 2004)
  – Background Knowledge: YAGO + REVERB

• Ranked knowledge are checked by three annotators
  – Totally annotated 7,000 background knowledge
  – When confliction happens, we choose the label selected by more annotators
  – Compute Fleiss’ Kappa value between annotators to evaluate annotation consistency
    • Best situation: 0.8066
    • On average: 0.7076
Experiments

• Data Preparation
  – Source document: ACE corpus (Doddington et al., 2004)
  – Background Knowledge: YAGO + REVERB
• Ranked knowledge are checked by three annotators
  – Totally annotated 7,000 background knowledge
  – When confliction happens, we choose the label selected by more annotators
  – Compute Fleiss’ Kappa value between annotators to evaluate annotation consistency
    • Best situation: 0.8066
    • On average: 0.7076

GOOD CONSISTENCE!
Experiments

• Overview
  – Baseline: compute relevance between *background knowledge* and source document by accumulating relevance to *sd-nodes*
Experiments

• Overview
  – Baseline: compute relevance between background knowledge and source document by accumulating relevance to sd-nodes
Experiments

- Different Setups
  - The effect of automatic extraction of source document
Experiments

- Different Setups
  - The effect of automatic extraction of *background knowledge*

![Bar chart showing comparison between Auto-extracted and YAGO for MAP and P&N metrics. The chart indicates higher values for Auto-extracted in MAP and lower values in P&N compared to YAGO.]
Experiments

- Different Setups
  - The effect of automatic extraction of *background knowledge*

![Graph showing performance metrics for MAP and P&N with two categories: Auto-extracted and YAGO.](image-url)
CONCLUSION & FUTURE WORK
Conclusion & Future Work

• There are always some knowledge gaps in documents
• Our model finds relevant background knowledge from multiple sources for a certain source document
• Our model extracts source documents and background knowledge automatically -- useful in real applications
Conclusion & Future Work

- There are always some knowledge gaps in documents
- Our model finds relevant background knowledge from multiple sources for a certain source document
- Our model extracts source documents and background knowledge automatically -- useful in real applications

- To further improve the ranking performance
- Automatic evaluation, instead of manual annotation
- To apply these background knowledge in real tasks

2014/8/26 Tuesday
Thanks

Q&A