Probabilistic, Structure-Aware Algorithms for Improved Variety, Accuracy, and Coverage of AMR Alignments

Austin Blodgett & Nathan Schneider
Georgetown University
@ ACL 2021
Linguistically Enriched AMR Aligner
Contributions

- A novel *structurally-comprehensive* formulation of AMR-to-English alignment in terms of mappings between spans and connected subgraphs.
- **Released Data**: automatic data for AMR Release 3.0 and *Little Prince* data + 350 gold aligned sentences.
- **Alignment algorithm** which combines EM with rules. Advantages include:
  1. much improved *coverage* over previous datasets,
  2. increased *variety* of the substructures aligned, including alignments for all relations, and alignments for diagnosing reentrancies,
  3. alignments are made between *spans* and *connected substructures* of an AMR,
  4. broader *identification of spans* including named entities and verbal and prepositional multiword expressions.
Outline

1. AMR Alignment: Background
2. LEAMR: Multi-layer Formulation
3. Released Data
4. Algorithm
5. Results
Introduction: AMR

• Abstract Meaning Representation captures “who did what to whom”
Introduction: AMR

- Abstract Meaning Representation captures “who did what to whom”

Directed acyclic graph representation of sentence meaning
Introduction: AMR

• Abstract Meaning Representation captures “who did what to whom”

Directed acyclic graph representation of sentence meaning

Consolidates a number of semantic prediction task:
• word sense disambiguation
• semantic role labelling
• named entity recognition
• coreference
Introduction: AMR

• Abstract Meaning Representation captures “who did what to whom”

Directed acyclic graph representation of sentence meaning

Consolidates a number of semantic prediction task:
• word sense disambiguation
• semantic role labelling
• named entity recognition
• coreference

Scalable (~60,000 available English sentences)
Introduction: AMR

• Abstract Meaning Representation captures “who did what to whom”

Directed acyclic graph representation of sentence meaning

Consolidates a number of semantic prediction task:
  • word sense disambiguation
  • semantic role labelling
  • named entity recognition
  • coreference

Scalable (~60,000 available English sentences)

Unanchored (lacks gold alignments)
“Most of the students want to visit New York when they graduate”
Background: Alignment

- Alignment in MT
e.g., German-to-English

- AMR Alignment
e.g., AMR-to-English
Alignments in AMR Parsing

Some AMR parsers rely on alignments: Composition-based parsers (e.g., Beschke, 2019; Lindemann et al., 2020; Groschwitz, 2019), transition-based parsers (Wang et al., 2015; Zhou et al., 2021; Astudillo et al., 2020; Naseem et al., 2019), factorization-based parsers (Flanigan et al., 2014)

For other AMR parsers (Lyu & Titov, 2018; Bevilacqua et al., 2021; Xu et al., 2020; Zhang et al., 2019), explicit alignments could still be valuable for evaluation.
Previous AMR Aligners

- **Rule Based:** JAMR alignments (Flanigan et al., 2014) align using iterative application of a list of rules.

- **Expectation-Maximization:** ISI alignments (Pourdamghani et al., 2014) first linearize an AMR and then apply an expectation-maximization alignment.

- **Tuned Alignments:** TAMR alignments (Liu et al., 2018) are built on top of the JAMR alignment system, but are tuned based on the performance of an oracle.

- **Graph Distance:** Wang and Xue (2017) use an HMM-based aligner and include a calculation of graph distance as a locality constraint, similar to our use of projection distance.
Limitations of Previous Aligners

Alignments are generally between individual nodes and individual tokens without full coverage:

- Nodes in an alignment may be disconnected
- Lack of multi-token alignments
- Non-comprehensive node coverage
- Low coverage and performance on edges
- No alignment of reentrancies

<table>
<thead>
<tr>
<th></th>
<th>nodes</th>
<th>edges</th>
<th>reentrancies</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAMR</td>
<td>91.1</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>ISI</td>
<td>78.7</td>
<td>9.8</td>
<td>✗</td>
</tr>
<tr>
<td>TAMR</td>
<td>94.9</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
Most of the students want to visit New York when they graduate.
Most of the students want to visit New York when they graduate.
Most of the students want to visit New York when they graduate.
Most of the students want to visit New York when they graduate.
Most of the students want to visit New York when they graduate.
Most of the students want to visit New York **when** they graduate.

**Relation Layer**

**used for:**
- argument structures,
- prepositions,
- etc.

**Alignments**
- **of**: \( :\text{ARG}0 \rightarrow :\text{ARG}1 \rightarrow :\text{ARG}2 \rightarrow :\text{ARG}3 \)
- **want**: \( :\text{ARG}0 \rightarrow :\text{ARG}1 \)
- **visit**: \( :\text{ARG}0 \rightarrow :\text{ARG}1 \)
- **when**: \( \rightarrow :\text{time} \)
- **graduate**: \( \rightarrow :\text{ARG}0 \)
Most of the students want to visit New York when they graduate.
Most **of** the students **want** to **visit** New York **when** they **graduate**
LEAMR Released Data

Automatic Alignments:
● AMR Release 3.0
● Little Prince

Gold Alignments:
● 350 sentences

<table>
<thead>
<tr>
<th>IAA</th>
<th>Exact Align F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subgraphs (366)</td>
<td>94.54</td>
</tr>
<tr>
<td>Relations (260)</td>
<td>90.73</td>
</tr>
<tr>
<td>Reentrancies (65)</td>
<td>76.92</td>
</tr>
<tr>
<td>Duplicates (5)</td>
<td>66.67</td>
</tr>
</tbody>
</table>

https://github.com/ablodge/leamr
Structure-Aware EM Algorithm

For alignable elements (unaligned nodes or edges) of the graph, do until finished:

- Identify legal candidate spans
  - unaligned spans
  - spans aligned to a neighboring element
  - (for subgraphs only) any span aligned to a duplicate of this element
- Score each candidate based on alignment and distance probabilities
- Align best scoring element-span pair
Structure-Aware EM Algorithm

For alignable elements (unaligned nodes or edges) of the graph, do until finished:

- Identify legal candidate spans
  - unaligned spans
  - spans aligned to a neighboring element
  - (for subgraphs only) any span aligned to a duplicate of this element
- Score each candidate based on alignment and distance probabilities
- Align best scoring element-span pair
Structure-Aware EM Algorithm

For alignable elements (unaligned nodes or edges) of the graph, do until finished:

- Identify legal candidate spans
  - unaligned spans
  - spans aligned to a neighboring element
  - (for subgraphs only) any span aligned to a duplicate of this element

- Score each candidate based on alignment and distance probabilities

- Align best scoring element-span pair
Structure-Aware EM Algorithm

For alignable elements (unaligned nodes or edges) of the graph, do until finished:

- Identify legal candidate spans
  - unaligned spans
  - spans aligned to a neighboring element
  - (for subgraphs only) any span aligned to a duplicate of this element
- Score each candidate based on alignment and distance probabilities
- Align best scoring element-span pair

alignments

\[
\begin{align*}
\text{span}_4 & \rightarrow \text{n}_0 \\
\text{span}_3 & \rightarrow \text{n}_1, \text{n}_2
\end{align*}
\]
Structure-Aware EM Algorithm

For alignable elements (unaligned nodes or edges) of the graph, do until finished:

- Identify legal candidate spans
  - unaligned spans
  - spans aligned to a neighboring element
  - (for subgraphs only) any span aligned to a duplicate of this element
- Score each candidate based on alignment and distance probabilities
- Align best scoring element-span pair

```
alignments
span_4 -> n_0
span_3 -> n_1, n_2
span_3 -> n_3
```

```
span_0 span_1 span_2 span_3 span_4 span_5 span_6 span_7 span_8 span_9 ...
```
For alignable elements (unaligned nodes or edges) of the graph, do until finished:

- Identify legal candidate spans
  - unaligned spans
  - spans aligned to a neighboring element
  - (for subgraphs only) any span aligned to a *duplicate* of this element
- Score each candidate based on alignment and distance probabilities
- Align best scoring element-span pair

**Structure-Aware EM Algorithm**

![Diagram of the structure-aware EM algorithm with labeled spans: span₀, span₁, span₂, span₃, span₄, span₅, span₆, span₇, span₈, span₉, ...]
Projection Distance

For two neighboring elements (nodes or edges), we define projection distance as the \textit{signed} distance between spans aligned to each element.

The \textcolor{red}{\textbf{old}} house\textsubscript{1} is bigger than the \textcolor{blue}{\textbf{new}} house\textsubscript{2}.
Projection Distance

For two neighboring elements (nodes or edges), we define projection distance as the signed distance between spans aligned to each element.

The *old* house$_1$ is bigger than the *new* house$_2$. 
Aligning Subgraphs

\[
\text{score}(\langle g, s \rangle) = P_{\text{align}}(g \mid s; \theta_1) \cdot \prod_{d_i \in D} P_{\text{dist}}(d_i; \theta_2)^{\frac{1}{|D|}} \cdot IB(g, s)
\]

- subgraph label given span label
- projection distance probability
- inductive bias

New York
Aligning Relations

\[
score(\langle a, s \rangle) = P_{\text{align}}(a \mid s; \theta_3) \cdot \prod_{d_i \in D_1} P_{\text{dist}}(d_i; \theta_4)^{\frac{1}{|D_1|}} \cdot \prod_{d_j \in D_2} P_{\text{dist}}(d_j; \theta_5)^{\frac{1}{|D_2|}}
\]

- reentrancy label given span label
- projection distance probability (parent)
- projection distance probability (child)
Aligning Reentrancies

\[ \text{score}(\langle r, s, \text{type} \rangle) = P_{\text{align}}(r, \text{type} | s; \theta_6) \cdot P_{\text{dist}}(d_1; \theta_7) \cdot P_{\text{dist}}(d_2; \theta_8) \]

- reentrancy label given span label
- projection distance probability (parent)
- projection distance probability (child)
DATA + CODE:

https://github.com/ablodge/leamr

Other AMR research:
https://nert-nlp.github.io/AMR-Bibliography/

Thank You!