UKP-BIU: Similarity and Entailment Metrics for Student Response Analysis

Torsten Zesch† **Omer Levy**§ **Iryna Gurevych**† **Ido Dagan**§

† Ubiquitous Knowledge Processing Lab
Computer Science Department
Technische Universität Darmstadt

§ Natural Language Processing Lab
Computer Science Department
Bar-Ilan University

Abstract

Our system combines text similarity measures with a textual entailment system. In the main task, we focused on the influence of lexicalized versus unlexicalized features, and how they affect performance on unseen questions and domains. We also participated in the pilot partial entailment task, where our system significantly outperforms a strong baseline.

1 Introduction

The Joint Student Response Analysis and 8th Recognizing Textual Entailment Challenge (Dzikovska et al., 2013) brings together two important dimensions of Natural Language Processing: real-world applications and semantic inference technologies. The challenge focuses on the domain of middle-school quizzes, and attempts to emulate the meticulous marking process that teachers do on a daily basis. Given a question, a reference answer, and a student's answer, the task is to determine whether the student answered correctly. While this is not a new task in itself, the challenge focuses on employing textual entailment technologies as the backbone of this educational application. As a consequence, we formalize the question "Did the student answer correctly?" as "Can the reference answer be inferred from the student's answer?". This question can (hopefully) be answered by a textual entailment system (Dagan et al., 2009).

The challenge contains two tasks: In the **main task**, the system must analyze each answer as a whole. There are three settings, where each one defines "correct" in a different resolution. The highest-resolution setting defines five different classes or "correctness values": correct, partially correct, contradictory, irrelevant, non-domain. In the **pilot task**, critical elements of the answer need to be analyzed separately. Each such element is called a *facet*, and is defined as a pair of words that are critical in answering the question. As there is a substantial difference between the two tasks, we designed sibling architectures for each task, and divide the main part of the paper accordingly.

Our goal is to provide a robust architecture for student response analysis, that can generalize and perform well in multiple domains. Moreover, we are interested in evaluating how well general-purpose technologies will perform in this setting. We therefore approach the challenge by combining two such technologies: **DKPro Similarity** –an extensive suite of text similarity measures– that has been successfully applied in other settings like the SemEval 2012 task on semantic textual similarity (Bär et al., 2012a) or reuse detection (Bär et al., 2012b).

BIUTEE, the Bar-Ilan University Textual Entailment Engine (Stern and Dagan, 2011), which has shown state-of-the-art performance on recognizing textual entailment challenges. Our systems use both technologies to extract features, and combine them in a supervised model. Indeed, this approach works relatively well (with respect to other entries in the challenge), especially in unseen domains.

2 Background

2.1 Text Similarity

Text similarity is a bidirectional, continuous function which operates on pairs of texts of any length and returns a numeric score of how similar one text is to the other. In previous work (Mihalcea et al.,

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 285–289, Atlanta, Georgia, June 14-15, 2013.

2006; Gabrilovich and Markovitch, 2007; Landauer et al., 1998), only a single text similarity measure has typically been applied to text pairs. However, as recent work (Bär et al., 2012a; Bär et al., 2012b) has shown, text similarity computation can be much improved when a variety of measures are combined.

In recent years, UKP lab at TU Darmstadt has developed DKPro Similarity[1], an open source toolkit for analyzing text similarity. It is part of the DKPro framework for natural language processing (Gurevych et al., 2007). DKPro Similarity excels at the tasks of measuring semantic textual similarity (STS) and detecting text reuse (DTR), having achieved the best performance in previous challenges (Bär et al., 2012a; Bär et al., 2012b).

2.2 Textual Entailment

The *textual entailment* paradigm is a generic framework for applied semantic inference (Dagan et al., 2009). The most prevalent task of textual entailment is to recognize whether the meaning of a target natural language statement (H for hypothesis) can be inferred from another piece of text (T for text). Apparently, this core task underlies semantic inference in many text applications. The task of analyzing student responses is one such example. By assigning the student's answer as T and the reference answer as H, we are basically asking whether one can infer the correct (reference) answer from the student's response. In recent years, Bar-Ilan University has developed BIUTEE (Stern and Dagan, 2011), an extensive textual entailment recognition engine. BIUTEE tries to convert T (represented as a dependency tree) to H. It does so by applying a series of knowledge-based transformations, such as synonym substitution, active-passive conversion, and more. BIUTEE is publicly available as open source.[2]

3 Main Task

In this section, we explain how we approached the main task, in which the system needs to analyze each answer as a whole. After describing our system's architecture, we explain how we selected training data for the different scenarios in the main task. We then provide the details for each submitted run, and finally, our empirical results.

3.1 System Description

We build a system based on the Apache UIMA framework (Ferrucci and Lally, 2004) and DKPro Lab (Eckart de Castilho and Gurevych, 2011). We use DKPro Core[3] for preprocessing. Specifically, we used the default DKPro segmenter, TreeTagger POS tagger and chunker, Jazzy Spell Checker, and the Stanford parser.[4] We trained a supervised model (Naive Bayes) using Weka (Hall et al., 2009) with feature extraction based on clearTK (Ogren et al., 2008). The following features have been used:

BOW features Bag-of-word features are based on the assumption that certain words need to appear in a correct answer. We used a mixture of token unigrams, bigrams, and trigrams, where each n-gram is a binary feature that can either be true or false for a document.[5] Additionally, we also used the number of tokens in the student answer as another feature in this group.

Syntactic Features We extend BOW features with syntactic functions by adding dependency and phrase n-grams. Dependency n-grams are combinations of two tokens and their dependency relation. Phrase n-grams are combinations of the main verb and the noun phrase left and right of the verb. In both cases, we use the 10 most frequent n-grams.

Basic Similarity Features This group of features computes the similarity between the reference answer and the student answer. In case there is more than one reference answer, we compute all pairwise similarity scores and add the minimum, maximum, average, and median similarity.[6]

Semantic Similarity Features are very similar to the basic similarity features, except that we use semantic similarity measures in order to bridge a possible vocabulary gap between the student and reference answer. We use the ESA measure (Gabrilovich

[1] code.google.com/p/dkpro-similarity-asl

[2] cs.biu.ac.il/~nlp/downloads/biutee

[3] code.google.com/p/dkpro-core-asl/

[4] DKPro Core v1.4.0, TreeTagger models v20130204.0, Stanford parser PCFG model v20120709.0.

[5] Using the 750 most frequent n-grams gave good results on the training set, so we also used this number for the test runs.

[6] As basic similarity measures, we use greedy string tiling (Wise, 1996) with $n = 3$, longest common subsequence and longest common substring (Allison and Dix, 1986), and word n-gram containment(Lyon et al., 2001) with $n = 2$.

and Markovitch, 2007) based on concept vectors build from WordNet, Wiktionary, and Wikipedia.

Spelling Features As spelling errors might be indicative of the answer quality, we use the number of spelling errors normalized by the text length as an additional feature.

Entailment Features We run BIUTEE (Stern and Dagan, 2011) on the test instance (as T) with each reference answer (as H), which results in an array of numerical entailment confidence values. If there is more than one reference answer, we compute all pairwise confidence scores and add the minimum, maximum, average, and median confidence.

3.2 Data Selection Regime

There are three scenarios under which our system is expected to perform. For each one, we chose (a-priori) a different set of examples for training.

Unseen Answers Classify new answers to familiar questions. Train on instances that have the same question as the test instance.

Unseen Questions Classify new answers to unseen (but related) questions. Partition the questions according to their IDs, creating sets of related questions, and then train on all the instances that share the same partition as the test instance.

Unseen Domains Classify new answers to unseen questions from unseen domains. Use all available training data from the same dataset.

3.3 Submitted Runs

The runs represent the different levels of lexicalization of the model which we expect to have strong influence in each scenario:

Run 1 uses all features as described above. We expect the BOW features to be helpful for the *Unseen Answers* setting, but to be misleading for unseen questions or domains, as the same word indicating a correct answer for one question might correspond to a wrong answer for another question.

Run 2 uses only non-lexicalized features, i.e. all features except the BOW and syntactic features, in order to assess the impact of the lexicalization that overfits on the topic of the questions. We expect this run to be less sensitive to the topic changes in the *Unseen Questions* and *Unseen Domains* settings.

Run 3 uses only the basic similarity and the entailment features. It should indicate the baseline performance that can be expected without targeting the system towards a certain topic.

Task	Run	Unseen Answers	Unseen Questions	Unseen Domains
2-way	1	**.734**	**.678**	.671
	2	.665	.644	.677
	3	.662	.625	.677
3-way	1	**.670**	**.573**	.572
	2	.595	.561	.577
	3	.574	.540	.576
5-way	1	**.590**	.376	**.407**
	2	.495	**.397**	.371
	3	.461	.394	.376

Table 1: Main task performance for the SciEntsBank test set. We show weighted average F_1 for the three scenarios.

	Cor.	Par	Con.	Irr.	Non.
Correct	**903**	463	164	309	78
Partially Correct	219	**261**	93	333	80
Contradictory	61	126	**91**	103	36
Irrelevant	209	229	119	**476**	189
Non-Domain	0	0	0	2	**18**

Table 2: Confusion matrix of *Run 1* in the 5-way *Unseen Domains* scenario. The vertical axis is the real class, the horizontal axis is the predicted class.

3.4 Empirical Results

Table 1 shows the F_1-measure (weighted average) of the three runs. As was expected for the *Unseen Answers* scenario, *Run 1* using a lexicalized model outperformed the other two runs. However, in the other scenarios *Run 1* is not significantly better, as lexicalized features do not have the same impact if the question or the domain changes.

Table 2 shows the confusion matrix of *Run 1* in the 5-way *Unseen Domains* scenario. The *Correct* category was classified quite reliably, but the *Irrelevant* category was especially hard. While the reference answer provides some clues for what is correct or incorrect, the range of things that are "irrelevant" for a given question is potentially very big and thus cannot be easily learned. We also see that the system ability to distinguish *Correct* and *Partially Correct* answers need to be improved.

It is difficult to provide an exact assessment of our system's performance (with respect to other systems in the challenge), since there are multiple tasks, sce-

narios, datasets, and even metrics. However, we can safely say that our system performed above average in most settings, and showed competitive results in the *Unseen Domains* scenario.

4 Pilot Task

In the pilot task each facet needs to be analyzed separately, which requires some changes in the system architecture.

4.1 System Description

We segmented and lemmatized the input data using the default DKPro segmenter and the TreeTagger lemmatizer. The partial entailment system is composed of three methods: *Exact*, *WordNet*, and *BIUTEE*. These were combined in different combinations to form the different runs.

Exact In this baseline method, we represent a student answer as a bag-of-words containing all tokens and lemmas appearing in the text. Lemmas are used to account for minor morphological differences, such as tense or plurality. We then check whether both facet words appear in the set.

WordNet checks whether both facet words, or their semantically related words, appear in the student's answer. We use WordNet (Fellbaum, 1998) with the Resnik similarity measure (Resnik, 1995) and count a facet term as matched if the similarity score exceeds a certain threshold (0.9, empirically determined on the training set).

BIUTEE processes dependency trees instead of bags of words. We therefore added a pre-processing stage that extracts a path in the dependency parse that represents the facet. This is done by first parsing the entire reference answer, and then locating the two nodes mentioned in the facet. We then find their lowest common ancestor (LCA), and extract the path from the facet's first word to the second through the LCA. BIUTEE can now be given the student answer and the pre-processed facet, and try to recognize whether the former entails the latter.

4.2 Submitted Runs

In preliminary experiments using the provided training data, we found that the very simple *Exact Match* baseline performed surprisingly well, with 0.96 precision and 0.32 recall on positive class instances (expressed facets). We therefore decided to use this feature as an initial filter, and employ the others for classifying the "harder" cases. Training BIUTEE only on these cases, while dismissing easy ones, improved our system's performance significantly.

	Unseen Answers	Unseen Questions	Unseen Domains
Baseline	.670	.688	.731
Run 1	.756	.710	.760
Run 2	**.782**	**.765**	**.816**
Run 3	.744	.733	.770

Table 3: Pilot task performance across different scenarios. The scores are F_1-measures (weighted average).

Run 1: *Exact* OR *WordNet* This is essentially just the *WordNet* feature on its own, because every instance that *Exact* classifies as positive is also positive by *WordNet*.

Run 2: *Exact* OR (*BIUTEE* AND *WordNet*) If the instance is non-trivial, this configuration requires that both *BIUTEE* and *WordNet Match* agree on positive classification. Equivalent to the majority rule.

Run 3: *Exact* OR *BIUTEE* *BIUTEE* increases recall of expressed facets at the expense of precision.

4.3 Empirical Results

Table 3 shows the F_1-measure (weighted average) of each run in each scenario, including *Exact Match* as a quite strong baseline. In the majority of cases, *Run 2* that combines entailment and WordNet-based lexical matching, significantly outperformed the other two. It is interesting to note that the systems' performance does not degrade in "harder" scenarios; this is a result of the non-lexicalized nature of our methods. Unfortunately, our system was the only submission in this track, so we do not have any means to perform relative comparison.

5 Conclusion

We combined semantic textual similarity with textual entailment to solve the problem of student response analysis. Though our features were not tailored for this task, they proved quite indicative, and adapted well to unseen domains. We believe that additional generic features and knowledge resources are the best way to improve on our results, while retaining the same robustness and generality as our current architecture.

Acknowledgements

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806, and by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287923 (EXCITEMENT). We would like to thank the Minerva Foundation for facilitating this cooperation with a short term research grant.

References

Lloyd Allison and Trevor I. Dix. 1986. A bit-string longest-common-subsequence algorithm. *Information Processing Letters*, 23:305–310.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012a. UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the 6th International Workshop on Semantic Evaluation and the 1st Joint Conference on Lexical and Computational Semantics*, pages 435–440, June.

Daniel Bär, Torsten Zesch, and Iryna Gurevych. 2012b. Text reuse detection using a composition of text similarity measures. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 167–184, December.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2009. Recognizing textual entailment: Rationale, evaluation and approaches. *Natural Language Engineering*, 15(4):i–xvii.

Myroslava O. Dzikovska, Rodney Nielsen, Chris Brew, Claudia Leacock, Danilo Giampiccolo, Luisa Bentivogli, Peter Clark, Ido Dagan, and Hoa Trang Dang. 2013. Semeval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge. In **SEM 2013: The First Joint Conference on Lexical and Computational Semantics*, Atlanta, Georgia, USA, 13-14 June. Association for Computational Linguistics.

Richard Eckart de Castilho and Iryna Gurevych. 2011. A lightweight framework for reproducible parameter sweeping in information retrieval. In *Proceedings of the 2011 workshop on Data infrastructurEs for supporting information retrieval evaluation (DESIRE '11)*, New York, NY, USA. ACM.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

David Ferrucci and Adam Lally. 2004. UIMA: An architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3-4):327–348.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 1606–1611.

Iryna Gurevych, Max Mühlhäuser, Christof Müller, Jürgen Steimle, Markus Weimer, and Torsten Zesch. 2007. Darmstadt Knowledge Processing Repository based on UIMA. In *Proceedings of the 1st Workshop on Unstructured Information Management Architecture at Biannual Conference of the Society for Computational Linguistics and Language Technology*, Tübingen, Germany, April.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1).

Thomas K. Landauer, Peter W. Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes*, 25(2&3):259–284.

Caroline Lyon, James Malcolm, and Bob Dickerson. 2001. Detecting short passages of similar text in large document collections. In *Proceedings of the 6th Conference on Empirical Methods in Natural Language Processing (EMNLP 2001)*, pages 118–125, Pittsburgh, PA USA.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 775–780, Boston, MA.

Philip V. Ogren, Philipp G. Wetzler, and Steven Bethard. 2008. ClearTK: A UIMA Toolkit for Statistical Natural Language Processing. In *Towards Enhanced Interoperability for Large HLT Systems: UIMA for NLP workshop at Language Resources and Evaluation Conference (LREC)*.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 1995)*, pages 448–453.

Asher Stern and Ido Dagan. 2011. A confidence model for syntactically-motivated entailment proofs. In *Proceedings of the 8th International Conference on Recent Advances in Natural Language Processing (RANLP 2011)*, pages 455–462.

Michael J. Wise. 1996. YAP3: Improved detection of similarities in computer program and other texts. In *Proceedings of the 27th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 1996)*, pages 130–134, Philadelphia, PA.