# Visual Error Analysis for Entity Linking

**Benjamin Heinzerling**
Research Training Group AIPHES
Heidelberg Institute for
Theoretical Studies gGmbH
Schloss-Wolfsbrunnenweg 35
69118 Heidelberg, Germany
benjamin.heinzerling@h-its.org

**Michael Strube**
Heidelberg Institute for
Theoretical Studies gGmbH
Schloss-Wolfsbrunnenweg 35
69118 Heidelberg, Germany
michael.strube@h-its.org

## Abstract

We present the Visual Entity Explorer (VEX), an interactive tool for visually exploring and analyzing the output of entity linking systems. VEX is designed to aid developers in improving their systems by visualizing system results, gold annotations, and various mention detection and entity linking error types in a clear, concise, and customizable manner.

## 1 Introduction

Entity linking (EL) is the task of automatically linking mentions of entities (e.g. persons, locations, organizations) in a text to their corresponding entry in a given knowledge base (KB), such as Wikipedia or Freebase. Depending on the setting, the task may also require detection of entity mentions[1], as well as identifying and clustering Not-In-Lexicon (NIL) entities.

In recent years, the increasing interest in EL, reflected in the emergence of shared tasks such as the TAC Entity Linking track (Ji et al., 2014), ERD 2014 (Carmel et al., 2014), and NEEL (Cano et al., 2014), has fostered research on evaluation metrics for EL systems, leading to the development of a dedicated scorer that covers different aspects of EL system results using multiple metrics (Hachey et al., 2014).

Based on the observation that representations in entity linking (mentions linked to the same KB entry) are very similar to those encountered in coreference resolution (mentions linked by coreference relations to the same entity), these metrics include ones originally proposed for evaluation of coreference resolutions systems, such as the MUC score (Vilain et al., 1995), $B^3$ (Bagga and Baldwin, 1998), and *CEAF* (Luo, 2005) and variants thereof (Cai and Strube, 2010).

While such metrics, which express system performance in numeric terms of precision, recall, and *F1* scores, are well-suited for comparing systems, they are of limited use to EL system developers trying to identify problem areas and components whose improvement will likely result in the largest performance increase.

To address this problem, we present the Visual Entity Explorer (VEX), an interactive tool for visually exploring the results produced by an EL system. To our knowledge, there exist no other dedicated tools for visualizing the output of EL systems or similar representations.

VEX is available as free, open-source software for download at `http://github.com/noutenki/vex` and as a web service at `http://cosyne.h-its.org/vex`.

In the remainder of this paper, we first give an overview of VEX (Section 2), proceed to present several usage examples and discuss some of the insights gained from performing a visual error analysis (Section 3), then describe its implementation (Section 4), before concluding and discussing future work (Section 5).

## 2 The Visual Entity Explorer

After loading system results and gold standard annotations in TAC 2014 or JSON format, as well as the original document text files, VEX displays

[1]This setting is called *Entity Discovery and Linking* (EDL) in the TAC 2014/15 entity linking tracks, and *Entity Recognition and Disambiguation* (ERD) in the Microsoft ERD 2014 challenge.



*Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 37–42,
Beijing, China, July 26-31, 2015. 


Figure 1: Screenshot of VEX's main display, consisting of document list (left), entity selectors (bottom right), and the annotated document text (top right).

gold annotations, correct results, and errors as shown in Figure 1. The document to be analyzed can be selected via the clickable list of document IDs on the left. Located bottom right, the entity selectors for gold, true positive, and false positive entities (defined below) can be used to toggle the display of individual entities[2]. The selected entities are visualized in the top-right main area.

Similarly to the usage in coreference resolution, where a cluster of mentions linked by coreference relations is referred to as an *entity*, we define *entity* to mean a cluster of mentions clustered either implicitly by being linked to the same KB entry (in case of non-NIL mentions) or clustered explicitly by performing NIL clustering (in case of NIL mentions).

[2]For space reasons, the entity selectors are shown only partially.

### 2.1 Visualizing Entity Linking Errors

Errors committed by an EL system can be broadly categorized into mention detection errors and linking/clustering errors. Mention detection errors, in turn, can be divided into partial errors and full errors.

#### 2.1.1 Partial Mention Detection Errors

A partial mention detection error is a system mention span that overlaps but is not identical to any gold mention span. In VEX, partial mention detection errors are displayed using red square brackets, either inside or outside the gold mention spans signified by golden-bordered rectangles (cf. the first and last mention in Figure 2).

#### 2.1.2 Full Mention Detection Errors

A full mention detection error is either (a) a system mention span that has no overlapping gold mention span at all, corresponding to a false positive (FP) detection, i.e. a precision error, or (b) a

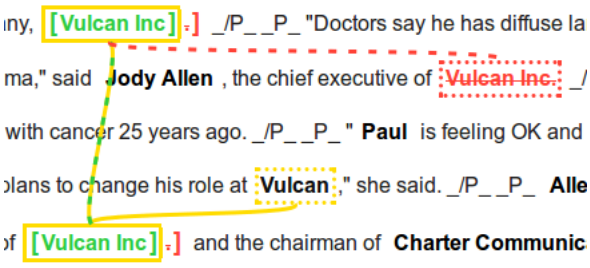


Figure 2: Visualization of various mention detection and entity linking error types (see Section 2 for a detailed description).

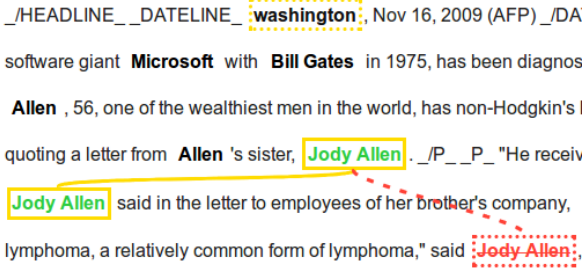


Figure 3: Visualization showing a mention detection error and an annotation error (see Section 3 for a description).

gold mention span that has no overlap with any system mention span, corresponding to a false negative (FN) detection, i.e. a recall error. In VEX, FP mention detections are marked by a dashed red border and struck-out red text (cf. the second mention in Figure 2), and FN mention detections by a dashed gold-colored border and black text (cf. the third mention in Figure 2). For further emphasis, both gold and system mentions are displayed in bold font.

#### 2.1.3 Linking/Clustering Errors

Entities identified by the system are categorized – and possibly split up – into True Positive (TP) and False Positive (FP) entities. The mentions of system entities are connected using dashed green lines for TP entities and dashed red lines for FP entities, while gold entity mentions are connected by solid gold-colored lines. This choice of line styles prevents loss of information through occlusion in case of two lines connecting the same pair of mentions, as is the case with the first and last mention in Figure 2.

Additionally, the text of system mentions linked to the correct KB entry or identified correctly as NIL is colored green and any text associated with erroneous system entity links red.

## 3 Usage examples

In this section we show how VEX can be used to perform a visual error analysis, gaining insights that arguably cannot be attained by relying only on evaluation metrics.

### 3.1 Example 1

Figure 2 shows mentions of VULCAN INC.[3] as identified by an EL system (marked red and green) and the corresponding gold annotation[4] (marked in gold color). Of the three gold mentions, two were detected and linked correctly by the system and are thus colored green and connected with a green dashed line. One gold mention is surrounded with a gold-colored dashed box to indicate a FN mention not detected by the system at all. The dashed red box signifies a FP entity, resulting from the system having detected a mention that is not listed in the gold standard. However, rather than a system error, this is arguably an annotation mistake.

Inspection of other entities and other documents reveals that spurious FPs caused by gold annotation errors appear to be a common occurrence (see Figure 3 for another example). Since the supervised machine learning algorithms commonly used for named entity recognition, such as Conditional Random Fields (Sutton and McCallum, 2007), require consistent training data, such inconsistencies hamper performance.

### 3.2 Example 2

From Figure 2 we can also tell that two mention detection errors are caused by the inclusion of sentence-final punctuation that doubles as abbreviation marker. The occurrence of similar cases in other documents, e.g. inconsistent annotation of "U.S." and "U.S" as mentions of UNITED STATES, shows the need for consistently applied annotation guidelines.

### 3.3 Example 3

Another type of mention detection error is shown in Figure 3: Here the system fails to detect "washington" as a mention of WASHINGTON, D.C.,

[3] In this paper, SMALL CAPS denote KB entries.

[4] The gold annotations are taken from the TAC 2014 EDL Evaluation Queries and Links (V1.1).

likely due to the non-standard lower-case spelling.

### 3.4 Example 4

The visualization of the gold mentions of PAUL ALLEN in Figure 1 shows that the EL system simplistically partitioned and linked the mentions according to string match, resulting in three system entities, of which only the first, consisting of the two "Paul Allen" mentions, is a TP. Even though the four "Allen" mentions in Figure 1 align correctly with gold mentions, they are categorized as a FP entity, since the system erroneously linked them to the KB entry for the city of Allen, Texas, resulting in a system entity that does not intersect with any gold entity. The system commits a similar mistake for the mention "Paul".

### 3.5 Insights

This analysis of only a few examples has already revealed several categories of errors, either committed by the EL system or resulting from gold annotation mistakes:

- mention detection errors due to non-standard letter case, which suggest incorporating truecasing (Lita et al., 2003) and/or a caseless named entity recognition model (Manning et al., 2014) into the mention detection process could improve performance;
- mention detection errors due to off-by-one errors involving punctuation, which suggest the need for clear and consistently applied annotation guidelines, enabling developers to add hard-coded, task-specific post-processing rules for dealing with such cases;
- mention detection errors due to missing gold standard annotations, which suggest performing a simple string match against already annotated mentions to find cases of unannotated mentions could significantly improve the gold standard at little cost;
- linking/clustering errors, likely due to the overly strong influence of features based on string match with Wikipedia article titles, which in some cases appears to outweigh features designed to encourage clustering of mentions if there exists a substring match between them, hence leading to an erroneous partitioning of the gold entity by its various surface forms.

## 4 Implementation

In this section we describe VEX's implementation and some of the design decisions made to achieve an entity visualization suited for convenient error analysis.

VEX consists of three main components. The `input` component, implemented in Java 8, reads gold and system annotations files, as well as the original documents. Currently, the annotation format read by the official TAC 2014 scorer[5], as well as a simple JSON input format are supported. All system and gold character offset ranges contained in the input files are converted into HTML spans and inserted into the document text. Since HTML elements are required to conform to a tree structure, any overlap or nesting of spans is handled by breaking up such spans into non-overlapping subspans.

At this point, gold NIL clusters and system NIL clusters are aligned by employing the Kuhn-Munkres algorithm[6] (Kuhn, 1955; Munkres, 1957), as is done in calculation of the *CEAF* metric (Luo, 2005). The `input` component then stores all inserted, non-overlapping spans in an in-memory database.

The `processing` component queries gold and system entity data for each document and inventorizes all errors of interest. All data collected by this component is added to the respective HTML spans in the form of CSS classes, enabling simple customization of the visualization via a plain-text stylesheet.

The `output` component employs a template engine[7] to convert the data collected by the `processing` component into HTML and JavaScript for handling display and user interaction in the web browser.

### 4.1 Design Decisions

One of VEX's main design goals is enabling the user to quickly identify entity linking and clustering errors. Because a naive approach to entity visualization by drawing edges between all possible pairings of mention spans quickly leads to a cluttered graph (Figure 4a), we instead visualize entities using Euclidean minimum spanning trees, inspired by Martschat and Strube's (2014) use of

[5] `http://github.com/wikilinks/neleval`

[6] Also known as Hungarian algorithm.

[7] `https://github.com/jknack/handlebars.java`


Figure 4: Cluttered visualization of an entity via its complete graph, drawing all pairwise connections between mentions (a), and a more concise visualization of the same entity using an Euclidean minimum spanning tree, connecting all mentions while minimizing total edge length (b).

spanning trees in error analysis for coreference resolution.

An Euclidean minimum spanning tree is a minimum spanning tree (MST) of a graph whose vertices represent points in a metric space and whose edge weights are the spatial distances between points[8], i.e., it spans all graph vertices while minimizing total edge length. This allows for a much more concise visualization (Figure 4b).

Since the actual positions of mention span elements on the user's screen depend on various user environment factors such as font size and browser window dimensions, the MSTs of displayed entities are computed using a client-side JavaScript library[9] and are automatically redrawn if the browser window is resized. Drawing of edges is performed via jsPlumb[10], a highly customizable library for line drawing in HTML documents.

In order not to overemphasize mention detection errors when displaying entities, VEX assumes a system mention span to be correct if it has a non-zero overlap with a gold mention span. For example, consider the first gold mention "Vulcan Inc" in Figure 2, which has not been detected correctly by the system; it detected "Vulcan Inc." instead. While a strict evaluation requiring perfect mention spans will give no credit at all for this partially correct result, seeing that this mention detection error is already visually signified (by the red square bracket), VEX treats the mention as detected correctly for the purpose of visualizing the entity graph, and counts it as a true positive instance if it has been linked correctly.

While VEX provides sane defaults, the visualization style can be easily customized via CSS, e.g., in order to achieve a finer-grained categorization of error types such as off-by-one mention detection errors, or classification of non-NILs as NILs and vice-versa.

## 5 Conclusions and Future Work

We presented the Visual Entity Explorer (VEX), a tool for visual error analysis of entity linking (EL) systems. We have shown how VEX can be used for quickly identifying the components of an EL system that appear to have a high potential for improvement, as well as for finding errors in the gold standard annotations. Since visual error analysis of our own EL system revealed several issues and possible improvements, we believe performing such an analysis will prove useful for other developers of EL systems, as well.

In future work, we plan to extend VEX with functionality for visualizing additional error types, and for exploring entities not only in a single document, but across documents. Given the structural similarities entities in coreference resolution and

[8] In our case, the metric space is the DOM document being rendered by the web browser, a point is the top-left corner of a text span element, and the distance metric is the pixel distance between the top-left corners of text span elements.

[9] `https://github.com/abetusk/euclideanmst.js`. This library employs Kruskal's algorithm (Kruskal, 1956) for finding MSTs.

[10] `http://www.jsplumb.org`

entities in entity linking share, we also will add methods for visualizing entities found by coreference resolution systems.

## Acknowledgements

This work has been supported by the German Research Foundation as part of the Research Training Group "Adaptive Preparation of Information from Heterogeneous Sources" (AIPHES) under grant No. GRK 1994/1, and partially funded by the Klaus Tschira Foundation, Heidelberg, Germany. We would like to thank our colleague Sebastian Martschat who commented on earlier drafts of this paper.

## References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the 1st International Conference on Language Resources and Evaluation,* Granada, Spain, 28–30 May 1998, pages 563–566.

Jie Cai and Michael Strube. 2010. Evaluation metrics for end-to-end coreference resolution systems. In *Proceedings of the SIGdial 2010 Conference: The 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue,* Tokyo, Japan, 24–25 September 2010, pages 28–36.

Amparo E. Cano, Giuseppe Rizzo, Andrea Varga, Matthew Rowe, Milan Stankovic, and Aba-Sah Dadzie. 2014. Making sense of microposts named entity extraction & linking challenge. In *Proceedings of the 4th Workshop on Making Sense of Microposts,* Seoul, Korea, 7 April 2014, pages 54–60.

David Carmel, Ming-Wei Chang, Evgeniy Gabrilovich, Bo-June Paul Hsu, and Kuansan Wang. 2014. ERD'14: Entity recognition and disambiguation challenge. In *ACM SIGIR Forum*, volume 48, pages 63–77. ACM.

Ben Hachey, Joel Nothman, and Will Radford. 2014. Cheap and easy entity evaluation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers),* Baltimore, Md., 22–27 June 2014, pages 464–469.

Heng Ji, Joel Nothman, and Ben Hachey. 2014. Overview of TAC-KBP2014 entity discovery and linking tasks. In *Proceedings of the Text Analysis Conference,* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 17–18 November 2014.

Joseph B. Kruskal. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50.

Harold W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Lucian Vlad Lita, Abe Ittycheriah, Salim Roukos, and Nanda Kambhatla. 2003. Truecasing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics,* Sapporo, Japan, 7–12 July 2003, pages 152–159. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the Human Language Technology Conference and the 2005 Conference on Empirical Methods in Natural Language Processing,* Vancouver, B.C., Canada, 6–8 October 2005, pages 25–32.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations,* Baltimore, Md., 22–27 June 2014, pages 55–60. Association for Computational Linguistics.

Sebastian Martschat and Michael Strube. 2014. Recall error analysis for coreference resolution. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing,* Doha, Qatar, 25–29 October 2014, pages 2070–2081.

James Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial & Applied Mathematics*, 5(1):32–38.

Charles Sutton and Andrew McCallum. 2007. An introduction to conditional random fields for relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*, pages 93–128. MIT Press, Cambridge, Mass.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th Message Understanding Conference (MUC-6)*, pages 45–52, San Mateo, Cal. Morgan Kaufmann.