Linkify: Enhanced Reading Experience by Augmenting Text Using Linked Open Data

Ikuya Yamada[1,2,3], Tomotaka Ito[1,2], Shinnosuke Usami[1,2], Shinsuke Takagi[1,2], Tomoya Toyoda[1,2], Hideaki Takeda[3], and Yoshiyasu Takefuji[2]

[1] Studio Ousia Inc., 4489-105-221 Endo, Fujisawa, Kanagawa, Japan
{ikuya,tomotaka,usa,shinny,tomoyan}@ousia.jp
[2] Keio University, 5322 Endo, Fujisawa, Kanagawa, Japan
takefuji@sfc.keio.ac.jp
[3] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda, Tokyo, Japan
takeda@nii.ac.jp

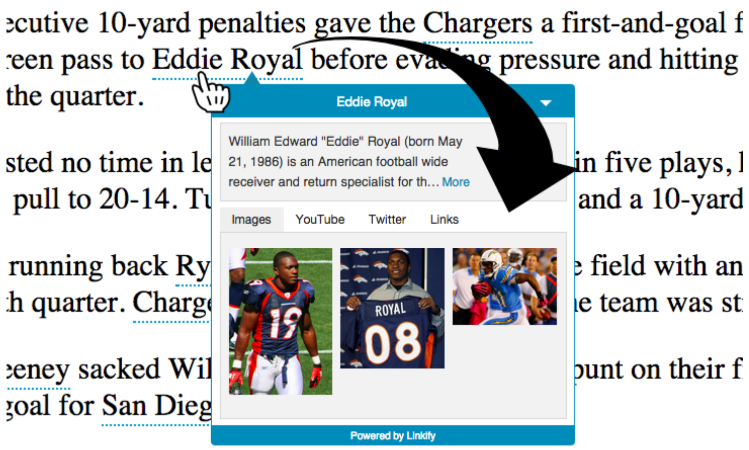
Abstract. We frequently encounter unfamiliar entity names (e.g., a person's name or a geographic location) while reading texts such as newspapers, magazines, and web pages. When it occurs, we typically perform a sequence of wearisome actions: select the entity name, submit it to a search engine, and typically obtain detailed information from web sites. In this paper, we propose *Linkify*, a novel tool that enhances text reading by automatically converting entity names into links and displaying a synopsis of the entity retrieved from *Linked Open Data* when a user selects the link. The tool enables users to retrieve the information of the entity simply by selecting the link. Further, in order to create only links that are helpful for users, we also developed a method that evaluates the helpfulness of entities using a machine-learning algorithm with a broad set of features. We have implemented our proposed tool as an add-on for several major web browsers and made it publicly available at `http://swc14.linkify.mobi`.

1 Introduction

In our daily lives, while reading text in newspapers, magazines, and web pages, we frequently encounter unfamiliar entities (e.g., a person's name or a geographic location). In such cases, we typically obtain related information from relevant web pages using a search engine. However, the process requires several time-consuming steps: select the entity name, submit a query to a search engine, and obtain detailed information from web sites.

In particular, these actions become especially problematic when a user is using touch-enabled devices such as smartphones and tablets, because merely selecting a span of text requires multiple actions, such as holding and swiping a finger on the screen. Further, current touch-screen devices are typically less accurate when using human fingers to specify a desired text position.

In this paper, we propose *Linkify*, a tool that automatically converts entity names within a document into links and displays summarized information about


ecutive 10-yard penalties gave the Chargers a first-and-goal f
een pass to Eddie Royal before evading pressure and hitting
the quarter.

sted no time in le
pull to 20-14. Tu
running back Ry
h quarter. Charg
eeney sacked Wil
goal for San Dieg

in five plays, l
and a 10-yard
e field with an
e team was st
ount on their f

William Edward "Eddie" Royal (born May 21, 1986) is an American football wide receiver and return specialist for th... More

Fig. 1. *Linkify* enhances text reading by automatically converting entity names into links. When a link is selected, a small widget containing summarized information of the entity is displayed.

the entity when a user selects the link (see Figure 1). Using Linkify, users can retrieve desired information about an unfamiliar entity simply by selecting the link, instead of having to select the text and submitting it to a search engine.

The tool uses *entity linking* internally to detect entity names. Entity linking is a method that involves automatic detection of the mentions of entities in a plain text and disambiguating them to the corresponding entries in a knowledge base. It has received considerable attention recently [2, 4] and has also been studied extensively [1, 3, 5–7]. We use a slightly modified version of a state-of-the-art entity linking method proposed by Milne and Witten [6]. In addition, *Wikipedia* is used as a knowledge base.

We also present an additional post-processing step that filters out *unhelpful* entities from the entities detected by the entity linking system. As current entity linking systems are typically targeted at usage as a component of other text analysis tasks, they typically attempt to detect all entities including entities that are not likely helpful for users. Wikipedia contains numerous unhelpful entities (e.g., *Japan*, *Monday*, *Computer*). Excessive creation of links can be distracting and typically degrades a user's overall experience[1]; therefore, this additional step is crucial. We use a method based on a machine-learning algorithm with a broad set of features which are mainly obtained from the *Linked Open Data* (LOD) cloud.

Further, we use an entity's class information (e.g., person, movie, geographic location) to display the summarized information that is displayed when the user selects an entity name. Using an entity's class, we can easily infer the typical user intention (e.g., *a map* for a geographic location, and *videos* for a movie name).

[1] Wikipedia instructs its contributers to insert links only where they are helpful.
`http://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style#Wikilinks`

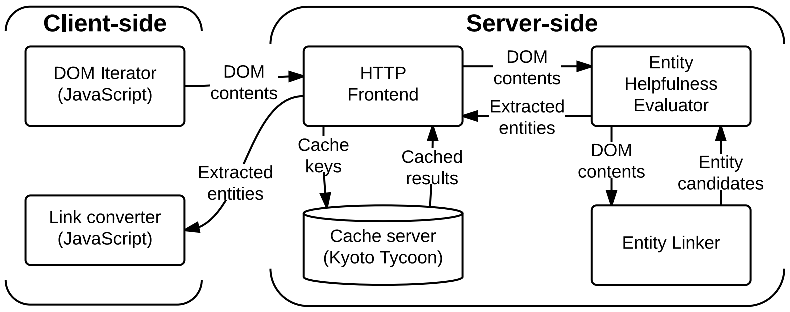


Fig. 2. Architecture of the proposed system.

Thus, we implement several pre-defined *content templates* and select the most appropriate template using rules written manually. At present, we use *DBpedia* to retrieve the entity classes.

Linkify currently runs on several major web browsers such as *Google Chrome*, *Mozilla Firefox*, and *Dolphin Browser*. It can also be easily integrated into existing web applications simply by inserting our small JavaScript code. The proposed system is a slightly modified version of our commercial version of Linkify[2], which has been provided stably for years.

2 The Proposed System

Our proposed system consists of two main components: a component for converting entity names into links, and another for rendering summarized information about an entity when a corresponding link is selected.

2.1 Converting Entity Names into Links

Figure 2 depicts an abstract architecture of our system that shows how it converts entity names in a document into links. The system comprises five modules: *DOM Iterator*, *HTTP Frontend*, *Entity Linker*, *Entity Helpfulness Evaluator*, and *Link Converter*. It operates as follows:

1. When the browser renders a web page, *DOM Iterator* gathers the page contents and sends them as a request to *HTTP Frontend*.
2. On receiving the request, *HTTP Frontend* checks whether the results are already in the cache. If they are not, it sends the request to *Entity Helpfulness Evaluator*.
3. *Entity Helpfulness Evaluator* first forwards the request to *Entity Linker*, assigns a probability to each entity that represents how likely the entity is to be helpful to users, and sends the results back to *Link Converter* via *HTTP Frontend*.
4. *Link Converter* then converts the extracted entity names into links if the corresponding probabilities are larger than a specified threshold.

[2] `http://www.linkify.mobi`

Client-side

The client-side system is implemented using a small JavaScript program comprising *DOM Iterator* and *Link Converter*.

DOM Iterator iterates over the DOM nodes in a web page. It starts from *BODY* and iterates over all the child DOM nodes by recursively walking within the DOM tree in a depth-first manner. In this step, we exclude several tags that do not contain visible text such as *HEAD*, *FORM*, and *SCRIPT*.

Link Converter converts the entity names into links. The module inserts an anchor tag around the entity names and adds an event handler that is triggered to render a widget.

Server-side

The server-side system is deployed on our cloud infrastructure. It is primarily implemented in Python, Cython[3], and C++, and comprises three separate components: *HTTP Frontend*, *Entity Helpfulness Evaluator*, and *Entity Linker*. Further, these components run on separate servers and can be scaled separately by adding cloud server instances. In addition, the internal communication between these components is implemented using our fast RPC implementation.[4]

HTTP Frontend is a simple HTTP server that also handles results caching. The system calculates the MD5 hash value for each of the DOM nodes and caches the result using the hash as the key. If the DOM node is not in the cache, the node text is sent to *Entity Helpfulness Evaluator*.

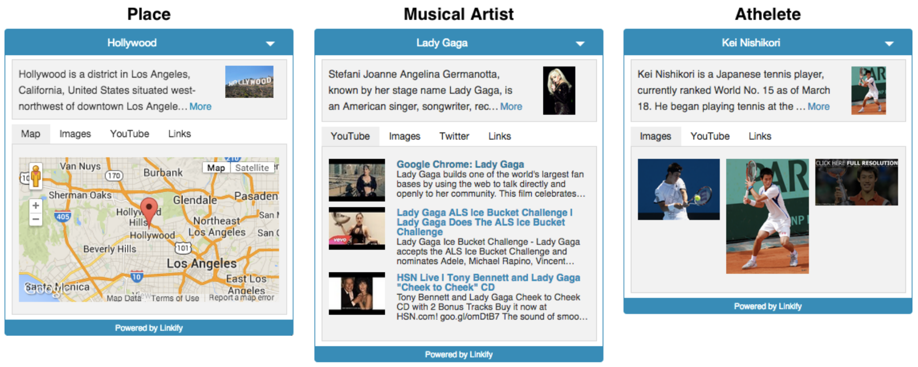
Entity Helpfulness Evaluator assigns a probability to each entity that specifies how likely the entity is to be helpful to users. We developed a novel algorithm that accurately evaluates the helpfulness of entities detected by the entity linking system. The algorithm formulates the problem as a binary classification task that detects whether the entity is likely helpful to users and utilizes *Random Forest* as a machine-learning algorithm. Further, we introduce a broad set of novel machine-learning features primarily extracted from the LOD cloud. (For further information, please refer to our previous work [8].)

Entity Linker extracts entities from the text and resolves them to the corresponding Wikipedia entries. We use a slightly modified version of a state-of-the-art entity linking method proposed by Milne and Witten [6].

2.2 Rendering a Widget

When the user selects a link, a small widget containing summarized information about the entity is displayed (see Figure 3). Entity classes are used to predict the user's intent and dynamically change the widget content.

[3] `http://cython.org/`
[4] `https://github.com/studio-ousia/mprpc`



Hollywood is a district in Los Angeles, California, United States situated west-northwest of downtown Los Angele... More

Stefani Joanne Angelina Germanotta, known by her stage name Lady Gaga, is an American singer, songwriter, rec... More

Google Chrome: Lady Gaga
Lady Gaga builds one of the world's largest fan bases by using the web to talk directly and openly to her community. This film celebrates...

Lady Gaga ALS Ice Bucket Challenge I Lady Gaga Does The ALS Ice Bucket Challenge
Lady Gaga Ice Bucket Challenge - Lady Gaga accepts the ALS Ice Bucket Challenge and nominates Adele, Michael Rapino, Vincent...

HSN Live I Tony Bennett and Lady Gaga "Cheek to Cheek" CD
Tony Bennett and Lady Gaga Cheek to Cheek CD with 2 Bonus Tracks Buy it now at HSN.com! goo.gl/omDtB7 The sound of smoo...

Kei Nishikori is a Japanese tennis player, currently ranked World No. 15 as of March 18. He began playing tennis at the ... More

Fig. 3. The widget displayed when a user selects a link. The most appropriate *content template* is automatically selected based on the entity class.

The widget is broadly divided into upper and lower halves. A short abstract and a thumbnail image describing the entity is shown in the upper half of the widget, while content changes dynamically according to the entity in the lower part of the widget.

The content types are as follows: *Map*: a simple Google map, *Twitter*: the timeline of the Twitter ID associated with the entity, *YouTube*: the search results using the entity name in YouTube, *Images*: search results obtained through Google Images, and *Links*: a list of related web pages retrieved from various web sources.

The content shown is chosen based on entity classes. According to the entity classes, the content is dynamically chosen via a set of manually written rules. For example, for an entity such as *Lady Gaga*, the YouTube content is chosen since the *Musical Artist* entity class is referred. Another example would be *Hollywood* as an entity. Since *Hollywood* belongs to the *Place* entity class, the Map would be chosen as the content in the lower part of the widget. In addition, we currently use *DBpedia Ontology Classes*[5] as the entity classes.

3 Using Linked Open Data

Our system is built on data collected from the LOD cloud. The data are effectively used for the following three different purposes:

The entity content displayed when the link is selected We use *DBpedia* and *Freebase* as our primary data sources. In order to generate the summarized information displayed in the widget, we use the entity's short descriptions, images, geographic coordinates, related web pages, and Twitter accounts.

Machine learning features for estimating entity helpfulness The main machine-learning features for estimating the helpfulness of entity names are

[5] `http://mappings.dbpedia.org/server/ontology/classes/`

extracted from the LOD cloud. In the machine-learning algorithm, we use a broad set of features extracted from DBpedia and Freebase such as statistics of the entity (e.g., the number of inbound links and the number of associated categories) and the classes of the entity.

Entity classes for selecting the relevant content template We use *DBpedia Ontology Classes* to select the most relevant *content template* for the selected entity. Using entity classes, the system can infer the most relevant content template from multiple predefined templates.

4 Conclusions

In this paper, we proposed Linkify, a tool that enhances text reading experience by augmenting the text using the data retrieved from the LOD cloud. The proposed system automatically converts entity names into links and displays a widget containing summarized information about the entity when a user selects the corresponding link. The system uses a significant amount of data retrieved from the LOD cloud.

This approach can further be recognized as building a *bridge* from the traditional web world to the emerging structured LOD world by automatically creating links in plain text to entities in the LOD cloud. This enables users to consume LOD resources effectively in their daily web experience.

References

1. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). pp. 708–716 (2007)
2. Ji, H., Grishman, R., Dang, H.T., Griffitt, K., Ellis, J.: Overview of the TAC 2010 knowledge base population track. In: Proceeding of Text Analytics Conference (TAC) (2010)
3. Kulkarni, S., Singh, A., Ramakrishnan, G., Chakrabarti, S.: Collective annotation of Wikipedia entities in web text. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09). pp. 457–466 (2009)
4. McNamee, P., Dang, H.: Overview of the TAC 2009 knowledge base population track. In: Proceeding of Text Analysis Conference (TAC) (2009)
5. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: DBpedia Spotlight: Shedding light on the web of documents. In: Proceedings of the 7th International Conference on Semantic Systems (I-Semantics '11). pp. 1–8 (2011)
6. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: Proceeding of the 17th ACM Conference on Information and Knowledge Management (CIKM '08). pp. 509–518 (2008)
7. Shen, W., Wang, J., Han, J.: Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. IEEE Transactions on Knowledge and Data Engineering 99(PrePrints), 1 (2014)

8. Yamada, I., Ito, T., Usami, S., Takagi, S., Takeda, H., Takefuji, Y.: Evaluating the helpfulness of linked entities to readers. In: Proceedings of the 25th ACM Conference on Hypertext and Social Media (HT '14). pp. 169–178 (2014)

Appendix

A Minimal requirements

The application has to be an end-user application: Our system is provided as an add-on for several major web browsers such as Google Chrome, Mozilla Firefox and Dolphin Browser. Further, web developers can easily integrate the proposed functionality into their applications by inserting our JavaScript code.

The information sources should be i) under diverse ownership, ii) heterogeneous and iii) contains real world data: We use data retrieved from heterogeneous data sources under diverse ownership. These sources include LOD repositories such as DBpedia and Freebase, and other sources such as Wikipedia, Twitter, Google Maps, and YouTube.

The meaning of data has to play a central role: Semantic Web technologies play a central role in our system. Our system is built upon data collected from the LOD cloud. The data are effectively used for the three purposes: 1) building summarized information that is displayed when the link is selected, 2) machine-learning features for estimating the helpfulness of entities, and 3) entity classes for selecting the most relevant content template used to render the widget. For further information, please see Section 3.

B Additional Desirable Features

The application provides an attractive and functional Web interface: We provide add-ons for major web browsers. Further, we carefully designed the appearance of our user interface with the utmost attention to the user experience.

The application should be scalable: Because our system detects helpful entities when the target text is rendered on the screen, speed is our primary concern. Our system is deployed on the cloud infrastructure and the components for detecting entity names can be scaled separately by adding cloud server instances. We currently use a total of 15 cloud instances, all having latest processors.

Rigorous evaluations have taken place: We extensively evaluated the machine-learning algorithm to evaluate the helpfulness of entities. We developed a dataset using a crowd-sourcing service and evaluated the algorithm from various aspects. The evaluation reveals that our algorithm significantly outperforms existing similar methods by *5.7%-12% F1*. For further information, please refer to our previous paper [8].

Further, we have publicly provided a commercial system for mobile developers to easily integrate the proposed functionality for years. The system has been successfully used by hundreds of developers and integrated into numerous real-world applications. This fact demonstrates the significance of our system.

Novelty, in applying semantic technology to a domain or task that have not been considered before: Only a few systems such as *DBpedia Spotlight* [5] and *Wikipedia Miner* [6], presently automatically create links on entity names. However, these systems only create direct links to the corresponding entries in knowledge bases such as DBpedia and Wikipedia. In contrast, our system has a novel feature that directly displays summarized information about an entity in a widget. It enables users to obtain relevant information without opening the link.

Further, to the best of our knowledge, our work is the first to specifically evaluate the helpfulness to users of entities in the document. As stated above, our machine-learning algorithm uses significant amounts of data retrieved from the LOD cloud.

Functionality is different from or goes beyond pure information retrieval: The proposed system significantly streamlines entity retrieval from a text by automatically converting entity names into links. The functionality inarguably goes beyond pure information retrieval.

Contextual information is used for ratings or rankings: When the system evaluates the helpfulness of an entity in the document, the algorithm considers whether the entity is coherent with the document topics. The basic idea is that if the entity is more related to the document topics, then it is likely to be more helpful to users. We measure this by calculating the average similarity score between the target entity and the other extracted entities. For further information, please see our previous paper [8].

Multimedia documents are used in some way: The proposed system displays images, maps, and videos retrieved from several web sources such as DBpedia, Freebase, Google Images, and YouTube.

There is a use of dynamic data (e.g. workflows), perhaps in combination with static information: The system displays summarized information about entities when users select a generated link. The data are retrieved from various web sources containing dynamically changing data such as Twitter and YouTube.

The results should be as accurate as possible: We use a novel machine-learning algorithm that accurately detects the most helpful entity names for users. Our method is significantly more accurate than similar methods [8]. Further, as explained above, we also use entity classes to infer desired information displayed in the widget.

There is support for multiple languages and accessibility on a range of devices: Our system can be executed in web browsers on multiple devices including desktop machines, smartphones and tablets.