From Zero to Hero: Human-In-The-Loop Entity Linking in Low Resource Domains

Jan-Christoph Klie **Richard Eckart de Castilho** **Iryna Gurevych**
Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science
Technical University of Darmstadt, Germany
www.ukp.tu-darmstadt.de

Abstract

Entity linking (EL) is concerned with disambiguating entity mentions in a text against knowledge bases (KB). It is crucial in a considerable number of fields like humanities, technical writing and biomedical sciences to enrich texts with semantics and discover more knowledge. The use of EL in such domains requires handling noisy texts, low resource settings and domain-specific KBs. Existing approaches are mostly inappropriate for this, as they depend on training data. However, in the above scenario, there exists hardly annotated data, and it needs to be created from scratch. We therefore present a novel domain-agnostic Human-In-The-Loop annotation approach: we use recommenders that suggest potential concepts and adaptive candidate ranking, thereby speeding up the overall annotation process and making it less tedious for users. We evaluate our ranking approach in a simulation on difficult texts and show that it greatly outperforms a strong baseline in ranking accuracy. In a user study, the annotation speed improves by 35 % compared to annotating without interactive support; users report that they strongly prefer our system. An open-source and ready-to-use implementation based on the text annotation platform INCEpTION[1] is made available[2].

1 Introduction

Entity linking (EL) describes the task of disambiguating entity mentions in a text by linking them to a knowledge base (KB), e.g. the text span *Earl of Orrery* can be linked to the KB entry *John Boyle, 5. Earl of Cork*, thereby disambiguating it. EL is highly beneficial in many fields like digital humanities, classics, technical writing or biomedical sciences for applications like search (Meij et al., 2014), semantic enrichment (Schlögl and Lejtovicz, 2017) or information extraction (Nooralahzadeh and Øvrelid, 2018). These are overwhelmingly low-resource settings: often, no data annotated exists; coverage of open-domain knowledge bases like Wikipedia or DBPedia is low. Therefore, entity linking is frequently performed against domain-specific knowledge bases (Munnelly and Lawless, 2018a; Bartsch, 2004).

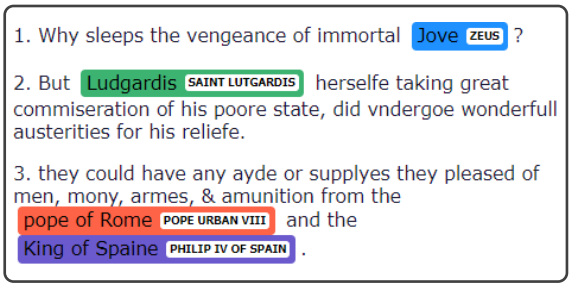

Figure 1: Difficult entity mentions with their linked entities: 1) Name variations, 2) Spelling Variation, 3) Ambiguity

In these scenarios, the first crucial step is to obtain annotated data. This data can then be either directly used by researchers for their downstream task or to train machine learning models for automatic annotation. For this initial data creation step, we developed a novel Human-In-The-Loop (HITL) annotation approach. Manual annotation is laborious and often prohibitively expensive. To improve annotation speed and quality, we therefore add interactive machine learning annotation support that helps the user find entities in the text and select the correct knowledge base entries for them. The more entities are annotated, the better the annotation support will be.

Throughout this work, we focus on texts from digital humanities, to be more precise, texts written in Early Modern English texts, including poems, biographies, novels as well as legal documents. In

[1] https://inception-project.github.io
[2] https://github.com/UKPLab/acl2020-interactive-entity-linking

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6982–6993
July 5 - 10, 2020.

this domain, texts are noisy as they were written in times where orthography was rather incidental or due to OCR and transcription errors (see Fig. 1). Tools like named entity recognizers are unavailable or perform poorly (Erdmann et al., 2019).

We demonstrate the effectiveness of our approach with extensive simulation as well as a user study on different, challenging datasets. We implement our approach based on the open-source annotation platform INCEpTION (Klie et al., 2018) and publish all datasets and code. Our contributions are the following:

1. We present a generic, KB-agnostic annotation approach for low-resource settings and provide a ready-to-use implementation so that researchers can easily annotate data for their use cases. We validate our approach extensively in a simulation and in a user study.

2. We show that statistical machine learning models can be used in an interactive entity linking setting to improve annotation speed by over 35%.

2 Related work

In the following, we give a broad overview of existing EL approaches, annotation support and Human-In-The-Loop annotation.

Entity Linking describes the task of disambiguating mentions in a text against a knowledge base. It is typically approached in three steps: 1) *mention detection*, 2) *candidate generation* and 3) *candidate ranking* (Shen et al., 2015) (Fig. 2). Mention detection most often relies either on gazetteers or pretrained named entity recognizers. Candidate generation either uses precompiled candidate lists derived from labeled data or uses full-text search. Candidate ranking assigns each candidate a score, then the candidate with the highest score is returned as the final prediction. Existing systems rely on the availability of certain resources like a large Wikipedia as well as software tools and often are restricted in the knowledge base they can link to. Off-the-shelf systems like `Dexter` (Ceccarelli et al., 2013), `DBPedia Spotlight` (Daiber et al., 2013) and `TagMe` (Ferragina and Scaiella, 2010) most often can only link against Wikipedia or a related knowledge base like Wikidata or DBPedia. They require good Wikipedia coverage for computing frequency statistics like popularity, view count or PageRank (Guo et al., 2013). These features work very well for standard datasets due to their Zipfian distribution of entities, leading to high reported scores on state-of-the art datasets (Ilievski et al., 2018; Milne and Witten, 2008). However, these systems are rarely applied out-of-domain such as in digital humanities or classical studies. Compared to state-of-the-art approaches, only a limited amount of research has been performed on entity linking against domain-specific knowledge bases. `AGDISTIS` (Usbeck et al., 2014) developed a knowledge-base-agnostic approach based on the HITS algorithm. The mention detection relies on gazetteers compiled from resources like Wikipedia and thereby performs string matching. Brando et al. (2016) propose `REDEN`, an approach based on graph centrality to link French authors to literary criticism texts. It requires additional linked data that is aligned with the custom knowledge base–they use DBPedia. As we work in a domain-specific low resource setting, access to large corpora which can be used to compute popularity priors is limited. We do not have suitable named entity linking tools, gazetteers or a sufficient amount of labeled training data. Therefore, it is challenging to use state of the art systems.

Human-in-the-loop annotation HITL machine learning describes an interactive scenario where a machine learning (ML) system and a human work together to improve their performance. The ML system gives predictions, and the human corrects if they are wrong and helps to spot things that have been overlooked by the machine. The system uses this feedback to improve, leading to better predictions and thereby reducing the effort of the human. In natural language processing, it has been applied in scenarios like interactive text summarization (Gao et al., 2018), parsing (He et al., 2016) or data generation (Wallace et al., 2019). Regarding machine-learning assisted annotation, Yimam et al. (2014) propose an annotation editor that during annotation, interactively trains a model using annotations made by the user. They use string matching and MIRA (Crammer and Singer, 2003) as recommenders, evaluate on POS and NER annotation and show improvement in annotation speed. `TASTY` (Arnold et al., 2016) is a system that is able to perform EL against Wikipedia on the fly while typing a document. A pretrained neural sequence tagger is being used that performs mention detection. Candidates are precomputed and the candidate is chosen that has the highest text sim-

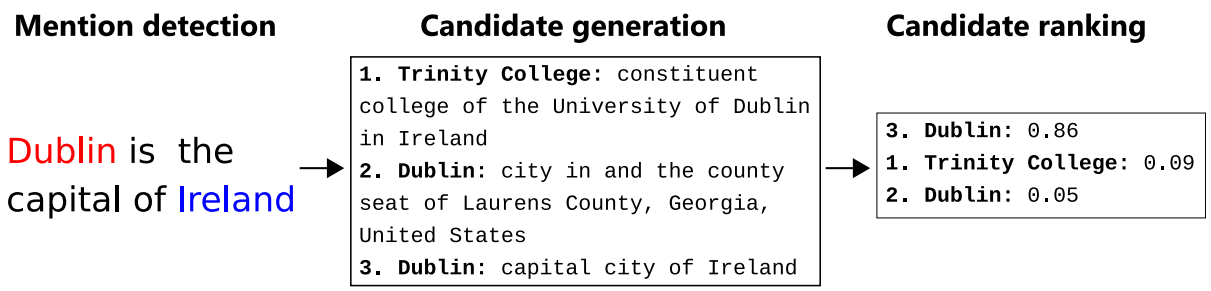

Figure 2: Entity linking pipeline: First, mentions of entities in the text need to be found. Then, given a mention, candidate entities are generated. Finally, entities are ranked and the top entity is chosen.

ilarity. The system updates its suggestions after interactions such as writing, rephrasing, removing or correcting suggested entity links. Corrections are used as training data for the neural model. However, due to the following reasons, it is not yet suitable for our scenario. In order to overcome the cold start problem, it needs annotated training data in addition to a precomputed index for candidate generation. It also only links against Wikipedia.

3 Architecture

The following section describes the three components of our annotation framework, following the standard entity linking pipeline (see Fig. 2). Throughout this work, we will mainly focus on the *candidate Ranking* step. We call the text span which contains an entity the *mention* and the sentence the mention is in the *context*. Each candidate from the knowledge base is assumed to have a label and a description. For instance, in Fig. 2, one mention is *Dublin*, the context is *Dublin is the capital of Ireland*, the label of the the first candidate is *Trinity College* and its description is *constituent college of the University of Dublin in Ireland*.

Mention Detection In the annotation setting, we rely on users to mark text spans that contain annotations. As support, we provide suggestions given by different recommender models: similar to Yimam et al. (2014), we use a string matcher suggesting annotations for mentions which have been annotated before. We also propose a new Levenshtein string matcher based on Levenshtein automata (Schulz and Mihov, 2002). In contrast to the string matcher, it suggests annotations for spans within a Levenshtein distance of 1 or 2. Preliminary experiments with ML models for mention detection like using a Conditional Random Field and handcrafted features did not perform well and yielded noisy suggestions, requiring further investigation.

Candidate Generation We index the knowledge base and use full text search to retrieve candidates based on the surface form of the annotated mention. Besides, users can query this index during annotation. We use fuzzy search to help in cases where the mention and the knowledge base label are almost the same but not identical (e.g. *Dublin* vs. *Dublyn*). In the interactive setting, the user can also search the knowledge base during annotation, e.g. in cases when the gold entity is not ranked high enough or when the surface form and knowledge base label are not the same (*Zeus* vs. *Jupiter*).

Candidate Ranking We follow Zheng et al. (2010) and model candidate ranking as a learning-to-rank problem: given a mention and a list of candidates, sort the candidates so that the most relevant candidate is at the top. For training, we guarantee that the gold candidate is present in the candidate list. For evaluation, the gold candidate can be absent from the candidate list if the candidate search failed to find it.

This interaction is the core Human-in-the-loop in our approach. For training, we rephrase the task as preference learning: By selecting an entity label from the candidate list, users express that the selected one was preferred over all other candidates. These preferences are used to train state-of-the-art pairwise learning-to-rank models from the literature: the gradient boosted trees variant `LightGBM` (Ke et al., 2017), `RankSVM` (Joachims, 2002) and `RankNet` (Burges et al., 2005). Models are retrained in the background when new annotations are made, thus improving over time with an increasing number of annotations. We use a set of generic handcrafted features which are described in Table 1. These models were chosen as they can work with low data, train quickly and allow introspection. Using deep models or word embeddings as input features showed to be too slow to be inter-

active. We also leverage pretrained Sentence-BERT embeddings (Reimers and Gurevych, 2019) trained on Natural Language Inference data written in simple English. These are not fine-tuned by us during training. Although they come from a different domain, we conjecture that the WordPiece tokenization of BERT helps with the spelling variance of our texts in contrast to traditional word embeddings which would have many out-of-vocabulary words. For specific tasks, custom features can easily be incorporated e.g. entity type information, time information for diachronic entity linking, location information or distance for annotating geographical entities.

• Mention exactly matches label
• Label is prefix/postfix of mention
• Mention is prefix/postfix of label
• Label is substring of mention and vice versa
• Levenshtein distance between mention and label
• Levenshtein distance between context and description
• Jaro-Winkler distance between mention and label
• Jaro-Winkler distance between context and description
• Sørensen-Dice index between context and description
• Jaccard coefficient between context and description
• Exact match of Soundex encoding of mention and label
• Phonetic Match Rating of mention and label
• Cosine distance between SBERT Embeddings of context and description (Reimers and Gurevych, 2019)
• Query length
∗ Query exactly matches label
∗ Query is prefix/postfix of label/mention
∗ Query is substring of mention/label
∗ Levenshtein distance between query and label
• Levenshtein distance between query and mention
• Jaro-Winkler distance between query and label
• Jaro-Winkler distance between query and mention

Table 1: Features used for candidate ranking. Starred features were also used by Zheng et al. (2010)

4 Datasets

There are very few datasets available that can be used for EL against domain-specific knowledge bases, further stressing our point that we need more of these, thereby requiring approaches like ours to create them. We use three datasets: *AIDA-YAGO*, *Women Writers Online* (WWO) and *1641 Depositions*. *AIDA* consists of Reuters news stories. To the best of our knowledge, WWO has not been considered for automatic EL so far. The 1641 Depositions have been used in automatic EL, but only when linking against DBPedia which has a very low entity coverage (Munnelly and Lawless, 2018b). We preprocess the data, split it in sentences, tokenize and reduce noise. For WWO, we derive a RDF KB from their personography, for 1641 we derive a knowledge base from the annotations. The exact processing steps as well as example texts are described in the appendix. The resulting data sets for WWO and *1641 Depositions* are also made available in the accompanying code repository.

AIDA-YAGO: For validating our approach, we evaluate on the AIDA-YAGO state-of-the art dataset introduced by Hoffart et al. (2011). Originally, this dataset is linked against YAGO and Wikipedia. We map the Wikipedia URLs to Wikidata and link against this KB, as Wikidata is available in RDF and the official Wikidata SPARQL endpoint offers full text search: it does not offer fuzzy search though.

Women Writers Online: *Women Writers Online*[3] is a collection of texts by pre-Victorian women writers. It includes texts on a wide range of topics and from various genres including poems, plays, and novels. They represent different states of the English language between 1400 and 1850. A subset of documents has been annotated with named entities (persons, works, places) (Melson and Flanders, 2010). Persons have also been linked to create a personography, a structured representation of persons' biographies containing names, titles, time and place of birth and death. The texts are challenging to disambiguate due to spelling variance, ciphering of names and a lack of standardized orthography. Sometimes, people are not referred to by name but by rank or function, e.g. *the king*. This dataset is interesting, as it contains documents with heterogeneous topics and text genres, causing low redundancy.

1641 Depositions: The *1641 Depositions*[4] contain legal texts in form of court witness statements recorded after the Irish Rebellion of 1641. In this conflict, Irish and English Catholics revolted against English and Scottish Protestants and their colonization of Ireland. It lasted over 10 years and ended with the Irish Catholics' defeat and the foreign rule of Ireland. The depositions have been transcribed from 17th century handwriting, keeping the old language and orthography. These documents have been used to analyze the rebellion, perform cold case reviews of the atrocities committed and to gain insights into contemporary life of this era. Part of the documents have been annotated

[3] https://www.wwp.northeastern.edu/wwo
[4] http://1641.tcd.ie/

Table 2: Data statistics of the three used datasets: Total number of **D**ocuments, **T**okens, **E**ntities, average number of **E**ntities per **S**entence, % of entities that are not linked. We also report the average number of entities linked to a mention, the average number of candidates when searching for a mention in the KB and the Gini coefficient which measures how balanced the entity distribution is.

Corpus	#D	#T	#E	#E/S	%NIL	Avg. Amb.	Avg. #Cand.	Gini
AIDA	1393	301,418	34,929	1.59	20.37	1.08	6.98	0.73
WWO	74	1,461,401	14,651	0.34	7.42	1.08	16.66	0.56
1641	16	11,895	480	2.40	0.0	1.01	36.29	0.44

with named entities that are linked to DBPedia (Munnelly and Lawless, 2018b). As the coverage of DBPedia was not sufficient (only around 20% of the entities are in DBPedia), we manually created a domain specific knowledge base for this data set containing places and people mentioned. To increase difficulty and reduce overfitting, we added additional related entities from DBPedia. The number of persons increases thereby by tenfold (130 → 1383) and the number of places by twentyfold (99 → 2119). Details for that can be found in Appendix A.1. While generating a KB from gold data is not ideal, creating or completing a knowledge base during annotation is not uncommon (see e.g. Wolfe et al., 2015). The texts are difficult to disambiguate due to the same reasons as for WWO. The depositions are interesting, as they contain documents from the same domain (witness reports), but feature many different actors and events.

Table 2 contains several statistics regarding the three datasets. AIDA and 1641 contain on average at least one entity per sentence, whereas WWO, while larger, is only sparsely annotated. In contrast to the other two, 1641 contains no entities linked to NIL. This is caused by the fact that we created the KB for 1641 from the gold annotations and for entities previously NIL, new entities were created by hand ; before that, the original corpus linking to DBPedia had 77% NIL annotations. The average ambiguity, that is, how many different entities were linked to mentions with the same surface form is quite high for AIDA and WWO and quite low for 1641. We explain the latter by the extreme variance in surface form, as even mentions of the same name are often written differently (e.g. *Castlekevyn* vs. *Castlekevin*). Also, 1641 contains many hapax legomena (mentions that only occur once). The average number of candidates is comparatively larger for WWO and 1641 as we use fuzzy search for these. Finally, the distributions of assigned entities in WWO and 1641 are also more balanced, expressed by a lower Gini coefficient (Dodge, 2008). These last two aspects together with noisy texts and low resources causes entity linking to be much more difficult compared to state-of-the-art datasets like AIDA.

5 Experiments

To validate our approach, we first evaluate recommender performance. Then, non-interactive ranking performance is evaluated similarly to state-of-the-art EL. Afterwards, we simulate a user annotating corpora with our Human-In-The-Loop ranker. Finally, we conduct a user study to test it in a realistic setting. Similar to other work on EL, our main metric for ranking is accuracy. We also measure Accuracy@5, as our experiments showed that users can quickly scan and select the right entity from a list of five elements. In our annotation editor, the candidate list shows the first five elements without scrolling. As a baseline, we use the Most-Frequently Linked Entity baseline (MFLEB). It assigns, given a mention, the entity that was most often linked to it in the training data.

5.1 Automatic suggestion performance

We evaluate the performance of our Levenshtein-based recommender that suggests potential annotations to users (Table 3). We filter out suggestions consisting of ≤ 3 characters as these introduce too much noise. For annotation suggestions, we focus on recall: where low precision implies recommendations that are not useful, no recall results in no recommendations at all. It can be seen that for AIDA and WWO, the performance of all three recommenders is quite good (recall is about 60% and 40%) while for 1641, it is only around 20%. The Levenshtein recommender increases recall and reduces precision. The impact is most pronounced for 1641, where it improves recall upon the string matching recommender by around 50%. In summary, we suggest using the string matching rec-

Dataset	Model	P	R	F1
AIDA	String	**0.43**	**0.60**	**0.50**
	Leven@1	0.31	0.55	0.40
	Leven@2	0.19	0.57	0.28
WWO	String	**0.17**	0.38	**0.23**
	Leven@1	0.11	0.40	0.16
	Leven@2	0.04	**0.42**	0.07
1641	String	0.12	0.14	0.13
	Leven@1	**0.16**	0.19	**0.17**
	Leven@2	0.12	**0.22**	0.15

Table 3: Recommender performance in **P**recision, **R**ecall and **F1** score for **String** matching recommender and **Leven**shtein recommender with distance 1 and 2. For AIDA, we evaluate on the test set, for the other datasets, we use 10-fold cross validation.

ommender for domains where texts are clean and exhibit low spelling variance. We consider the Levenshtein recommender to be more suitable for domains with noisy texts.

5.2 Candidate ranking performance

We evaluate EL candidate ranking in a non-interactive setting first to estimate the upper bound ranking performance. As we are the first to perform EL on our version of WWO and 1641, it also serves as a difficulty comparison between AIDA as the state-of-the-art dataset and datasets from our domain-specific setting. For AIDA, we use the existing train, development and test split; for the other two corpora, we perform 10-fold cross validation as we observed high variance in score when using different train-test splits. Features related to user queries are not used in this experiment. We assume that the gold candidate always exists in training and evaluation data. The results of this experiment are depicted in Table 4. It can be seen that for AIDA, the MFLE baseline is particularly strong, being better than all trained models. For the other datasets, the baseline is weaker than all, showing that popularity is a weak feature in our setting. For AIDA, LightGBM performs best, for WWO and 1641, the RankNet is best closely followed by the RankSVM. The accuracy@5 is comparatively high as there are cases where the candidate list is relatively short. Regarding training times, LightGBM trains extremely fast with RankSVM being a close second. They are fast enough to retrain after each user annotation. The RankNet trains two to four times slower than both.

Data	Model	A@1	A@5	$\lvert C \rvert$	t
AIDA	MFLEB	**0.56**	0.71	31	
	LightGBM	0.44	**0.72**		9
	RankSVM	0.37	0.69		56
	RankNet	0.42	0.70		190
WWO	MFLEB	0.32	0.77	19	
	LightGBM	0.37	0.83		2
	RankSVM	0.46	0.86		15
	RankNet	**0.52**	**0.87**		37
1641	MFLEB	0.28	0.75	38	
	LightGBM	0.35	0.77		1
	RankSVM	0.48	0.80		1
	RankNet	**0.55**	**0.83**		2

Table 4: Ranking scores when using all the data. We report **A**ccuracy@1 (Gold Candidate was ranked highest, **A**ccuracy@5 (Gold Candidate was in top 5 predictions of the ranker)). $|C|$ denotes the average number of candidates found for each mention. For AIDA, we evaluate on the test set, for the other datasets, we use 10-fold cross validation. We also measure the training time t in seconds averaged over 10 runs.

Feature importance The models we chose for ranking are white-box; they allow us to introspect the importance they give to each feature, thereby explaining their scoring choice. For the RankSVM, we follow Guyon et al. (2002) and use the square of the model weights as importance. For LightGBM, we use the number of times a feature is used to make a split in a decision tree. We train RankSVM and LightGBM models on all data and report the most important and least important features in Fig. 3. We normalize the weights by the L1-norm. It can be seen that both models rely on Levenshtein distance between mention and label as well as Sentence-BERT. The other text similarity features are, while sparingly, also used. Simple features like exact match, contains or prefix and postfix seem to not have a large impact. In general, LightGBM uses more features than the RankSVM. Even though Sentence-BERT was trained on Natural Language Inference (NLI) data which contains only relatively simple sentences, it still is relied on by both models for all datasets. The high importance of Levenshtein distance between mention and label for 1641 is expected and can be explained by the fact that the knowledge base labels often were derived from the mentions in the text when creating a domain-specific knowledge

base for this dataset. When trained on AIDA, the RankSVM assigns a high importance to the Jaccard distance between context and description. We attribute this to the fact that entity descriptions in Wikidata are quite short; if they are similar to the context then it is very likely a match.

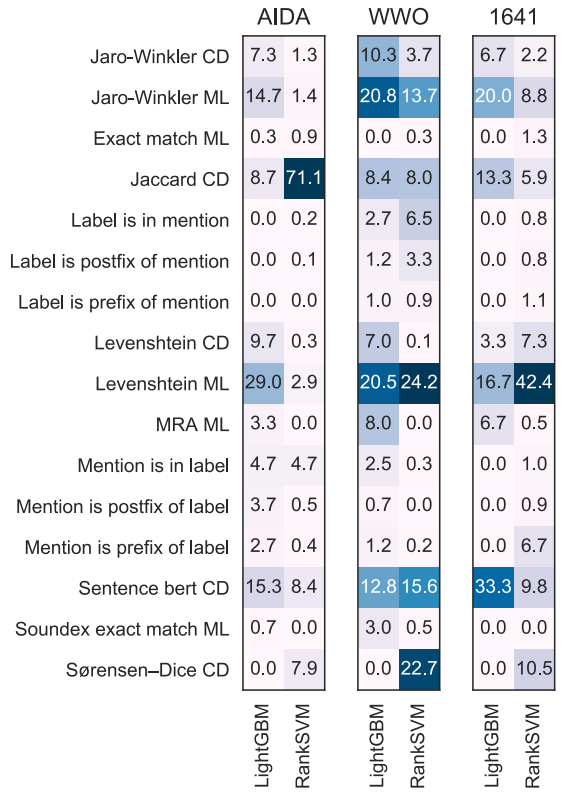

Figure 3: Feature importance of the respective models for different datasets. For the RankSVM, we use the squared weights; for LightGBM, we use the number of times a feature is used for splitting. Both are normalized to sum up to 1. ML stands for **M**ention-**L**abel, CD for **C**ontext-**D**escription.

5.3 Simulation

We simulate the Human-In-The-Loop setting by modeling a user annotating an unannotated corpus linearly. In the beginning, they annotate an initial seed of 10 entities without annotation support which are then used to bootstrap the ranker. At every step, the user annotates several entities where the ranker is used as assistance. After an annotation batch is finished, this new data is added to the training set, the ranker is retrained and evaluated. Only LightGBM and RankSVM are used as the RankNet turned out to be too slow. We do not evaluate on a holdout set. Instead, we follow Erdmann et al. (2019) and simulate annotating the complete corpus and evaluate on the very same data as we are interested in how an annotated subset helps to annotate the rest of the data, not how well the model generalizes. We assume that users annotate mention spans perfectly, i.e. we use gold spans. The candidate generation is simulated in three phases. It relies on the fact that the gold entity is given by the dataset: First, search for the mention only. If it was not found, search for the first word of the mention only. If this does not return the gold entity, search for the gold entity label. All candidates retrieved by these searches for a mention are used as training data. We also experimented with using only candidates for that the ranker assigned a higher score than the gold one. This, however, did not affect the performance. Therefore, we use all negative candidates.

Fig. 4 depicts the simulation results. All models outperform the MFLE baseline over most of the annotation process. It can be seen that both of our used models achieve high performance even if trained on very few annotations. The RankSVM handles low data better than LightGBM, but quickly reaches its peak performance due to it being a linear model with limited learning capacity. The LightGBM does not plateau that early. This potentially allows to first use a RankSVM for the cold start and when enough annotations are made, LightGBM, thereby combining the best of both models. Comparing the performance on the three datasets, we notice that the performance for AIDA is much higher. Also, the baseline rises much more steeply, hinting again that AIDA is easier and popularity there is a very strong feature. For 1641, the curve continue to rise, hinting that more data is needed to reach maximum performance.

Dataset	Phase 1	Phase 2	Phase 3
AIDA	0.20	0.00	0.80
WWO	0.26	0.27	0.47
1641	0.55	0.06	0.39

Table 5: Percentage of times the simulated user found the gold entity in the candidate list by searching for the mention (Phase 1), for the first word of the mention (Phase 2) or for the gold label (Phase 3).

Table 5 shows how the simulated user searched for the gold entities. We see that for WWO and 1641, the user often does not need to spend much effort in searching for the gold label, using the mention is in around 50% of the cases enough. We attribute this to the fuzzy search which the official Wikidata endpoint does not offer.

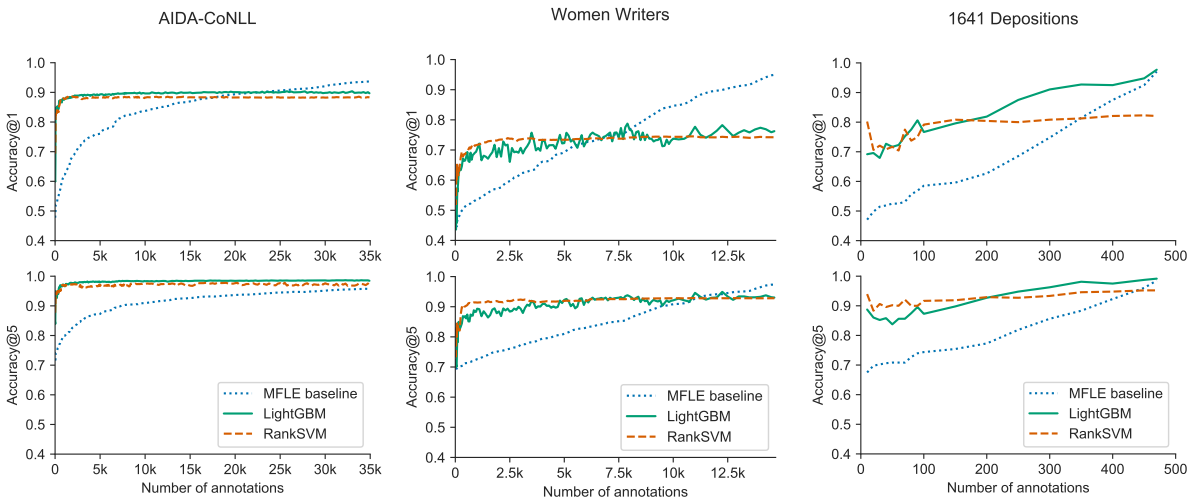

Figure 4: Human-in-the-loop simulation results for our three datasets and models. We can see that we get good Accuracy@5 with only a few annotations, especially for the `RankSVM`. This shows that the system is useful even at the beginning of the annotation process, alleviating the cold start problem.

5.4 User Study

In order to validate the viability of our approach in a realistic scenario, we conduct a user study. For that, we augmented the already existing annotation tool INCEpTION[5] (Klie et al., 2018) with our Human-In-The-Loop entity ranking and automatic suggestions. Fig. 5 shows a screenshot of the annotation editor itself. We let five users reannotate parts of the `1641` corpus. It was chosen as it has a high density of entity mentions while being small enough to be annotated in under one hour. Users stem from various academic backgrounds, e.g. natural language processing, computer science and digital humanities. Roughly half of them have previous experience with annotating. We compare two configurations: one uses our ranking and Levenshtein recommender, one uses the ranking of the full text search with the string matching recommender. We randomly selected eight documents which we split in two sets of four documents. To reduce bias, we assign users in four groups based on which part and which ranking they use first. Users are given detailed instructions and a warm-up document that is not used in the evaluation to get used to the annotation process. We measure annotation time, number of suggestions used and search queries performed. After the annotation is finished, we ask users to fill out a survey asking which system they prefer, how they experienced the annotation process and what suggestions they have to improve it. The evaluation of the user study shows that using our approach, users on average annotated 35% faster and needed 15% less search queries. Users positively commented on the ranking performance and the annotation suggestions for both systems. For our ranking, users reported that the gold entity often ranked first or close to top; they rarely observed that gold candidates were sorted close to the end of the candidate list.

We conduct a paired sample t-test to estimate the significance of our user study. Our null-hypothesis is that the reranking system does not improve the average annotation time. Conducting the test yields the following: $t = 3.332, p = 0.029$. We therefore reject the null hypothesis with $p = 0.029 < 0.05$, meaning that we have ample evidence that our reranking speeds up annotation time.

Recommender suggestions made up around 30% of annotations. We did not measure a significant difference between string and Levenshtein recommender. About the latter, users liked that it can suggest annotations for inexact matches. However, they criticized the noisier suggestions, especially for shorter mentions (e.g. annotating *joabe* (a name) yielded suggestions for *to be*). In the future, we will address this issue by filtering out more potentially unhelpful suggestions and using annotation rejections as a blacklist.

6 Conclusion

We presented a domain-agnostic annotation approach for annotating entity linking for low-resource domains. It consists of two main com-

[5] https://inception-project.github.io

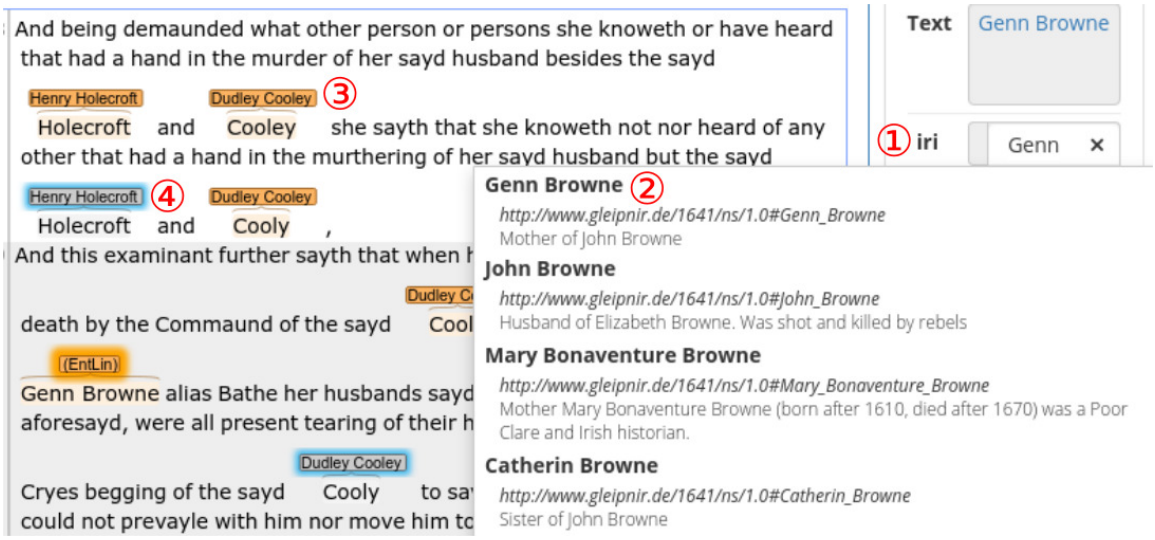

Figure 5: For our user study, we extend the INCEpTION annotation framework: ① entity linking search field, ② candidate list, ③ linked named entity, ④ entity linking recommendation.

ponents: recommenders that are algorithms that suggest potential annotations to users and a ranker that, given a mention span, ranks potential entity candidates so that they show up higher in the candidate list, making it easier to find for users. Both systems are retrained whenever new annotations are made, forming the Human-In-The-Loop.

Our approach does not require the existence of external resources like labeled data, tools like named entity recognizers or large-scale resources like Wikipedia. It can be applied to any domain, only requiring a knowledge base whose entities have a label and a description. In this paper, we evaluate on three datasets: `AIDA`, which is often used to validate state-of-the-art entity linking systems as well as `WWO` and `1641` from the humanities. We show that in simulation, only a very small subset needs to be annotated (fewer than 100) for the ranker to reach high accuracy. In a user study, results show that users prefer our approach compared to the typical annotation process; annotation speed improves by around 35% when using our system relative to using no reranking support.

In the future, we want to investigate more powerful recommenders, combine interactive entity linking with knowledge base completion and use online learning to leverage deep models, despite their long training time.

Acknowledgments

We thank the anonymous reviewers and Kevin Stowe for their detailed and helpful comments. We also want to thank the Women Writers Project which made the *Women Writers Online* text collection available to us. This work was supported by the German Research Foundation under grant № EC 503/1-1 and GU 798/21-1 as well as by the German Federal Ministry of Education and Research (BMBF) under the promotional reference 01UG1816B (CEDIFOR).

References

Sebastian Arnold, Robert Dziuba, and Alexander Löser. 2016. TASTY: Interactive Entity Linking As-You-Type. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 111–115.

Sabine Bartsch. 2004. Annotating a Corpus for Building a Domain-specific Knowledge Base. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, pages 1669–1672.

Carmen Brando, Francesca Frontini, and Jean-Gabriel Ganascia. 2016. REDEN: Named Entity Linking in Digital Literary Editions Using Linked Data Sets. *Complex Systems Informatics and Modeling Quarterly*, (7):60–80.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using Gradient Descent. In *Proceedings of the 22nd international conference on Machine learning - ICML '05*, pages 89–96.

Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. 2013. Dexter. In *Proceedings of the sixth international workshop on Exploiting semantic annotations in information retrieval - ESAIR '13*, pages 17–20.

Koby Crammer and Yoram Singer. 2003. Ultraconservative Online Algorithms for Multiclass Problems. *JMLR*, 3:951–991.

Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. 2013. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems - I-SEMANTICS '13*, pages 121–124.

Yadolah Dodge. 2008. *The Concise Encyclopedia of Statistics*. Springer.

Alexander Erdmann, David Joseph Wrisley, Benjamin Allen, Christopher Brown, Sophie Cohen-Bodénès, Micha Elsner, Yukun Feng, Brian Joseph, Béatrice Joyeux-Prunel, and Marie-Catherine de Marneffe. 2019. Practical, Efficient, and Customizable Active Learning for Named Entity Recognition in the Digital Humanities. In *Proceedings of the 2019 Conference of the North*, pages 2223–2234.

Paolo Ferragina and Ugo Scaiella. 2010. TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management - CIKM '10*, pages 1625–1628.

Yang Gao, Christian M. Meyer, and Iryna Gurevych. 2018. APRIL: Interactively Learning to Summarise by Combining Active Preference Learning and Reinforcement Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4120–4130.

Stephen Guo, Ming-Wei Chang, and Emre Kiciman. 2013. To Link or Not to Link? A Study on End-to-End Tweet Entity Linking. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1020–1030.

Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. 2002. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46:389–422.

Luheng He, Julian Michael, Mike Lewis, and Luke Zettlemoyer. 2016. Human-in-the-Loop Parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2337–2342.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of EMNLP'11*, pages 782–792.

Filip Ilievski, Piek Vossen, and Stefan Schlobach. 2018. Systematic Study of Long Tail Phenomena in Entity Linking. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 664–674.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '02*, pages 133–142.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9.

Edgar Meij, Krisztian Balog, and Daan Odijk. 2014. Entity linking and retrieval for semantic search. In *Proceedings of the 7th ACM international conference on Web search and data mining - WSDM '14*, pages 683–684.

John Melson and Julia Flanders. 2010. Not Just One of Your Holiday Games: Names and Name Encoding in the Women Writers Project Textbase. White paper, Women Writers Project, Brown University.

David Milne and Ian H. Witten. 2008. Learning to link with Wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge mining - CIKM '08*, pages 509–518.

Gary Munnelly and Séamus Lawless. 2018a. Constructing a knowledge base for entity linking on Irish cultural heritage collections. *Procedia Computer Science*, 137:199–210.

Gary Munnelly and Seamus Lawless. 2018b. Investigating Entity Linking in Early English Legal Documents. In *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries - JCDL '18*, pages 59–68.

Farhad Nooralahzadeh and Lilja Øvrelid. 2018. SIRIUS-LTG: An Entity Linking Approach to Fact Extraction and Verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 119–123.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3980–3990.

Matthias Schlögl and Katalin Lejtovicz. 2017. APIS - Austrian Prosopographical Information System. In *Proceedings of the Second Conference on Biographical Data in a Digital World 2017*.

Klaus U. Schulz and Stoyan Mihov. 2002. Fast string correction with Levenshtein automata. *International Journal on Document Analysis and Recognition*, 5(1):67–85.

Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.

Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Michael Röder, Daniel Gerber, Sandro Athaide Coelho, Sören Auer, and Andreas Both. 2014. AGDISTIS - Graph-Based Disambiguation of Named Entities Using Linked Data. In *The Semantic Web – ISWC 2014*, pages 457–471.

Eric Wallace, Pedro Rodriguez, Shi Feng, Ikuya Yamada, and Jordan Boyd-Graber. 2019. Trick Me If You Can: Human-in-the-loop Generation of Adversarial Question Answering Examples. *Transactions of the Association for Computational Linguistics*, 7(0):387–401.

Travis Wolfe, Mark Dredze, James Mayfield, Paul McNamee, Craig Harman, Tim Finin, and Benjamin Van Durme. 2015. Interactive Knowledge Base Population.

Seid Muhie Yimam, Chris Biemann, Richard Eckart de Castilho, and Iryna Gurevych. 2014. Automatic Annotation Suggestions and Custom Annotation Layers in WebAnno. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 91–96.

Zhicheng Zheng, Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010. Learning to Link Entities with Knowledge Base. In *Prooceedings of NAACL-HLT'10*, pages 483–491.

A Appendices

A.1 Dataset creation

The following section describes how we preprocess the raw texts from `WWO` and `1641`. Example texts can be found in Table 6. The respective code and datasets will be made available on acceptance.

A.1.1 Women Writers Online

We use the following checkout of the `WWO` data, which was graciously provided by the *Women Writers Project*[6].

```
Revision: 36425
Last Changed Rev: 36341
Last Changed Date: 2019-02-19
```

The texts itself are provided as TEI[7]. We use `DKPro Core`[8] to read in the TEI, split the raw text into sentences and tokenize it with the `JTokSegmenter`. When an annotation is spread over two sentences, we merge these sentences. This is mostly caused by a too eager sentence splitter. We covert the personographie which is in XML to RDF, including all properties that were encoded in there.

A.1.2 1641 Depositions

We use a subset of the 1641 depositions provided by Gary Munnelly. The raw data can be found on Github[9]. The texts itself are provided as NIF[10]. We use `DKPro Core`[11] to read in the NIF, split the raw text into sentences and tokenize it with the `JTokSegmenter`. When an annotation is spread over two sentences, we merge these sentences. This is mostly caused by a too eager sentence splitter. We use the knowledge base that comes with the NIF and create entities for all mentions that were `NIL`. We carefully deduplicate entities, e.g. `Luke Toole` and `Colonel Toole` are mapped to the same entity. In order to increase the difficulty of this dataset, we add additional entities from DBPedia: all Irish people, Irish cities and buildings in Ireland; all popes; royalities born between 1550 and 1650.

For that, we execute SPARQL queries against DBPedia for instances of `dbc:Popes`, `dbc:Royality`, `dbc:17th-century_Irish_people` and keep entries with a birth date before 1650 and a death date between 1600 and 1700. For the places, we search for `dbo:Castle`, `dbo:HistoricPlace`, `dbo:Building`, `dbc:17th-century_Irish_people` that are located in Ireland. The follwing table shows how many entities were in the original KB and how many were added:

Persons in gold data	130
Places in gold data	99
Persons added from DBPedia	1253
Places added from DBPedia	2020

[6] https://www.wwp.northeastern.edu/

[7] https://tei-c.org/

[8] https://dkpro.github.io/dkpro-core/

[9] https://github.com/munnellg/1641DepositionsCorpus

[10] https://persistence.uni-leipzig.org/nlp2rdf/

[11] https://dkpro.github.io/dkpro-core/

WWO
The following Lines occasion'd by the Marriage of Edward Herbert Esquire, and Mrs. Elizabeth Herbert. Cupid one day ask'd his Mother , When she meant that he shou'd Wed? You're too Young, my Boy, she said: Nor has Nature made another Fit to match with Cupid's Bed.
Finch, Anne: Miscellany poems, on several occasions, 1713
Joseph Joice of Kisnebrasney in the kings County gentleman sworne and examined deposeth and saith That after the Rebellion was begun in the County aforesaid vizt about the xxth of November 1641 This deponent for saffty fled to the Castle of knocknamease in the same County
Deposition of Joseph Joice, 1643[12]

Table 6: Example sentences from these corpora. Linked Named entities are highlighted in yellow.

A.2 Experiments

A.2.1 Full text search

For `AIDA` and Wikidata, we use the official SPARQL endpoint and the `Mediawiki API Query Service`[13]. It does not support fuzzy search. For `WWO` and `1641`, we host the created RDF in a Fuseki[14] instance and use the builtin functionality to index via Lucene.

A.2.2 Timing

Timing was performed on a Desktop PC with Ryzen 3600 and a GeForce RTX 2060.

[13] https://www.mediawiki.org/wiki/Wikidata_Query_Service/User_Manual/MWAPI

[14] https://jena.apache.org/documentation/fuseki2/