Evaluation of Incremental Entity Extraction with Background Knowledge and Entity Linking

Riccardo Pozzi
riccardo.pozzi@unimib.it
University of Milano-Bicocca
Milan, Italy

Federico Moiraghi
federico.moiraghimotta@unimib.it
University of Milano-Bicocca
SpazioDati S.r.l
Milan/Trento, Italy

Fausto Lodi
fausto.lodi@unimib.it
University of Milano-Bicocca
Milan, Italy

Matteo Palmonari
matteo.palmonari@unimib.it
University of Milano-Bicocca
Milan, Italy

ABSTRACT

Named entity extraction is a crucial task to support the population of Knowledge Bases (KBs) from documents written in natural language. However, in many application domains, these documents must be collected and processed incrementally to update the KB as more data are ingested. In some cases, quality concerns may even require human validation mechanisms along the process. While very recent work in the NLP community has discussed the importance of evaluating and benchmarking continuous entity extraction, it has proposed methods and datasets that avoid Named Entity Linking (NEL) as a component of the extraction process. In this paper, we advocate for batch-based incremental entity extraction methods that can exploit NEL with a background KB, detect mentions of entities that are not present in the KB yet (NIL mentions), and update the KB with the novel entities. Based on this assumption, we present a methodology to evaluate NEL-based incremental entity extraction, which can be applied to a "static" dataset for evaluating NEL into a dataset for evaluating incremental entity extraction. We apply this methodology to an existing benchmark for evaluating NEL algorithms, and evaluate an incremental extraction pipeline that orchestrates different strong state-of-the-art and baseline algorithms for the tasks involved in the extraction process, namely, NEL, NIL prediction, and NIL clustering. In presenting our experiments, we demonstrate the increased difficulty of the information extraction task in incremental settings and discuss the strengths of the available solutions as well as open challenges.

KEYWORDS

Incremental Entity Extraction, Entity Extraction, Knowledge Base Population, Named Entity Linking

IJCKG'22, October 27-29, 2022, Hangzhou, China (hybrid)

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Riccardo Pozzi, Federico Moiraghi, Fausto Lodi, and Matteo Palmonari. 2022. Evaluation of Incremental Entity Extraction with Background Knowledge and Entity Linking. In *The 11th International Joint Conference on Knowledge Graphs (IJCKG'22).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACKNOWLEDGMENTS

Part of this research has been funded by Cini in the context of the Italian project Datalake@Giustizia. We thank Lorenzo Sasso and Simone Monti for their valuable contributions.

1 INTRODUCTION

Updating a Knowledge Base (KB) - typically a Knowledge Graph - with information collected from textual documents is considered in several KB population methodologies to overcome the well-known population bottleneck [34]. Entity extraction is a key building block of these methodologies that can be supported at a large extent with Background KBs (BG-KB). In particular, *Named Entity Linking (NEL)* is the task of linking mentions of entities found in a document to their identifiers in a BG-KB. NEL is usually applied after entity mentions are identified in the text and eventually classified using some Named Entity Recognition (NER). While encyclopedic information sources, like Wikipedia and DBpedia, list and describe large collections of entities, they are far from being complete. Entities mentioned in a document that are not represented in the KB (NIL entities) must be identified as novel ones. Identifying mentions of novel entities is a sub-task of the process, often referred to as *NIL prediction*. After clustering NIL entity mentions that refer to the same real-world entity, a task usually referred to as *NIL clustering*, the KB can be updated with the new entities. NIL clustering is a task that is much related to *(cross-document) coreference resolution* [20] since both cluster together mentions referring the same real-world entities.

Combining NER, NEL, NIL prediction and NIL clustering algorithms makes it possible to execute pipelines for end-to-end *entity extraction with a BG-KB*, eventually extending the BG-KB with the novel entities found in the documents. Several methodologies, benchmark datasets, and tools have been proposed and used to evaluate solutions for entity extraction, in particular with a focus

on the NER task or on the combination of NEL, NIL prediction and NIL clustering [10, 13, 21, 27, 38].

However, the datasets proposed to evaluate NEL, NIL prediction, and NIL clustering assume that the entity extraction task is executed once on a given input corpus. Only recently it has been stressed that in many application scenarios entity extraction must be applied to a collection of documents that are ingested over time [4, 20], in a dynamic way. We refer to these scenarios as *incremental entity extraction*. In these scenarios, entity extraction must be performed on documents that are ingested incrementally in such a way that also the KB is extended incrementally exploiting *batches of documents* as they are ingested. In [20], authors propose a similar task, where entity coreference is applied to streams of documents, then they propose a benchmark for the evaluation, and discuss challenges that emerge in this continuous scenario. They explicitly contrast this approach to entity extraction to approaches that use NEL and BG-KBs. While we share the same arguments for motivating the need of entity extraction solutions that operate incrementally, we maintain that similar solutions can and should also be developed by exploiting BG-KBs, which provide valuable support, and thus consider the NEL, NIL prediction, and NIL clustering tasks in an incremental scenario. Human in the loop (HITL) in entity extraction is particularly relevant for ethical concerns when information from automatic systems is used to support decisions in sensitive application domains, such as in the juridical context [4]. In incremental entity extraction, we therefore assume that documents are ingested and processed in batches, in such a way that HITL may be incorporated at intermediate processing steps, may improve the quality of the extended KB.

Incremental entity extraction with BG-KB can be considered, as a whole, an incremental version of NEL (I-NEL), where NIL prediction and NIL clustering are follow-up tasks. As the process unfolds, we can consider the KB as consisting of two (virtual) parts: *BG-KB*, which may contain millions of entities since the beginning, and its extension, named *NEW-KB*. NEW-KB is empty at the beginning and updated incrementally as a new batch is processed: an example of incremental update is shown in figure 1. A key feature of I-NEL is that when processing the i-th batch, the NEL is expected to link not only to entities stored in the BG-KG, but also to entities stored in NEW-KB after processing previous batches. Thus, the NEL algorithm must use limited information when trying to link to entities stored in the NEW-KB after previous batches. The information about new entities may gradually increase, but errors can also propagate across batches. An example of the task is shown in figure 1, where the entity *John Smith* is not present in the BG-KB, however it is recognized as a new entity and added to the NEW-KB at the end of the batch processing; in the following batches, we expect that the mention of *John Smith* is correctly linked to the - now known - entity in the NEW-KB.

In addition to the definition of the *incremental entity extraction task with a BG-KB*, the main contributions of this paper are the following resources: 1) a methodology to evaluate the task by adapting benchmark datasets for entity extraction to the incremental (batch-based) scenario, 2) an incremental version of the WikilinksNED Unseen-Mentions (WNUM) [27], 3) a pipeline that combines strong baselines for each subtask, i.e., NEL, NIL prediction and NIL clustering, and 4) an evaluation of the pipeline and a discussion of the challenges introduced by the incremental scenario. The incremental version of WNUM, the solution to generate it, and the baseline pipeline are resources that are documented and made publicly available[1].

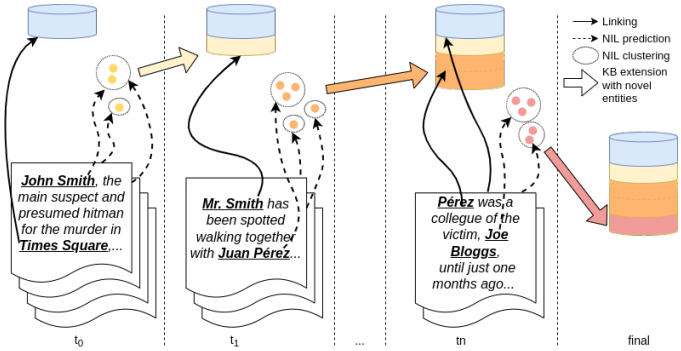

Figure 1: Documents are processed in batches through time; at each iteration, novel entities are added into the NEW-KB and can be linked in following steps. Between each step, a human validator can correct pipeline mistakes, splitting/merging clusters and fixing links.

The paper is organized as follows: we first discuss related work in section 2, then in section 3 we introduce a baseline pipeline to help the reader understand the task that is evaluated with the proposed methodology. Afterwards, we discuss the methodology to create the dataset (Section 4) and the result of its application to WNUM. Finally we discuss the evaluation on the incremental datasets (Section 5) and how the error propagates through the pipeline.

2 STATE OF THE ART

The conceptualization of the KB population with a BG-KG as a task composed of four sub-tasks is not novel and can be tracked back to the the TAC[2] (Text Analysis Conference), with its knowledge base population track (TAC-KBP), and to several other works [12]; but our contribution focus on an incremental version of this task, shortened as I-NEL. The importance of applying entity extraction solutions in an incremental setting has been recently stressed in prior works [20], which proposes methods without NEL. In this paper, we focus on presenting an *evaluation methodology* for the I-NEL task as well as an end-to-end *baselines*, which consider error propagation. In the present section, we first review recent work related to each subtask, to identify the best candidate components for the proposed pipelines; then we discuss limitations of current evaluation methodologies.

2.1 Named Entity Linking

The task of NEL consist in linking the correct entity e, taken from an arbitrary KB, to a given mention m. Recent works rely on neural networks of any kind, providing the most various strategies [29]. This work relies on the bi-encoder architecture [11], which uses BERT self-attention to map the mention vector into the significance

[1] https://github.com/rpo19/Incremental-Entity-Extraction
[2] https://tac.nist.gov/

space. This architecture, despite its simplicity, provides impressive results in both the candidate generation [35] and the candidate ranking step. It encodes separately mentions (in their context) and entities (given a short textual description) in the same space. For this reason, it is conceptually easy to obtain a (non-human) description of the entity, providing a sufficient number of examples, as seen in section 3.

In the last years, new and more complex models, always using self-attention mechanisms, obtained better performance on benchmark datasets [3, 6], at the price of introducing new problems such as output interpretability in evolving domains. Those models, in fact, rely only on meaningful entity names, without the need of a description. However, *"titles may not be enough"*[3]: names could be ambiguous in some cases, especially for entities that were not well-represented in either the train set or in the transformer pretraining. Other systems [19] instead ignore the problem of new entities, providing a simpler approach that embeds entities with a masked language model, which is constant in time.

The works presented above are only the first step in the pipeline, since it is followed by two additional components: the NIL predictor and the NIL cluster. Those components were often presented in previous works as future improvements, without a true experimentation. However, their presence can improve the whole pipeline, for both results checking and adding new entities into the KB.

Named Entity Linking systems are evaluated usually using micro-accuracy, calculating the fraction of mentions linked correctly, or macro-accuracy, averaging on the entities, thus calculating the fraction of entities linked correctly [9]. [35] is evaluated using micro-accuracy and also micro-recall@k (micro-recall@1 is equivalent to micro-accuracy), that corresponds to the percentage of mentions for which the correct entity is among the top-k ranked by BLINK.

2.2 NIL prediction

The NIL prediction task is often ignored by NEL systems that tend to omit NIL mentions from the datasets when evaluating and eventually leave this task to future works. Indeed, out of the 38 approaches compared by the survey [29] only 8 included NIL prediction. However, the TAC included NIL prediction in its KBP track starting from 2009 [23].

The task of NEL with NIL prediction can be seen as a classification task with a reject option that is NIL or unlinkable. There are four common approaches to perform NIL prediction [29]:

(1) Linking to NIL when the candidate generation gives an empty set of candidates.
(2) Setting a threshold for the linking score, below which a mention is considered unlinkable.
(3) Adding a NIL entity to the candidates' set, in the ranking phase. Hence, a new class is added to a classification problem.
(4) Training an additional binary classifier that accepts as input mention-entity pairs possibly with additional features, such as best linking score or information from NER, and that finally classifies whether a mention is linkable or not.

In this work we opt for the fourth approach, the binary classifier, as it allows exploiting additional features.

Being a classification task, NIL prediction is usually evaluated on precision, recall, and F1 score of the NIL class [9].

2.3 NIL clustering

The NIL clustering task is strongly related to *co-reference resolution*, since both try to group together mentions referring the same entities; however, NIL clustering does not have memory limitations (as far as the index is manageable) and uses as much information is available at a given time (documents are presented in batches of arbitrary size, and not one by one). Results, in this way, should be more accurate, since information is not removed from the index [20] and the output presents a mapping into the index itself (a problem ignored by the co-reference task [7]).

The problem since its introduction [25] has not been investigated exhaustively in literature: only 19 of the more than 150 solutions presented in [29] address the problem. Simple models [8, 15, 17, 18, 22, 31–33, 36, 37] address the problem using matches between surface forms, even using a fuzzy score given by a similarity criterion (Jaccard or edit-distance) or using custom rules (e.g. to manage acronyms). Some solutions [18, 32, 39] use unsupervised learning or supervised learning on custom features in order to group in clusters. Other approaches [2] instead exploits FastText [24] embeddings for clustering, or custom Word2Vec models [5] trained on languages that lacks pretrained language models, or address the problem with a three-steps solution [25]: mentions are grouped according to their surface form, then re-splitted according to a bag-of-words representation of documents and finally re-grouped together based on the centroid distance of the sub-clusters. As far as we know, no previous model uses embedding provided by a transformer architecture for clustering mentions.

Since the NIL clustering task is similar to coreference resolution, it can be evaluated on the same metrics: usually MUC, B^3, and CEAF [14, 26].

3 A BASELINE PIPELINE FOR INCREMENTAL ENTITY EXTRACTION

We propose a pipeline-based system composed by three modules: the NEL, the NIL prediction, and the NIL clustering modules. A schema of the pipeline is available in figure 2: given a mention with its context, the Linking module retrieves the best candidate for linking the mention, then the NIL prediction module detects if the best candidate is correct, in which case the mention is linked to the BG-KB. Otherwise, the mention is NIL and is given, along with all the other NIL mentions found in the current batch, to the NIL clustering module that clusters together those mentions referring to the same new entity. At this point, each cluster corresponds to a new entity to add to the NEW-KB so that it can be retrieved when linking future mentions. In this section, we further describe the pipeline components, and we finally discuss the representation of the novel entities.

3.1 Linking

The Linking module is based on the bi-encoder architecture [11], which is able to represent mentions (in the context) or entities (with a brief textual description) as dense vectors; at this point the linking score corresponds to the dot-product between mention and entity vectors.

F1	NIL	not-NIL	$\overline{F1}$
max	15.2	82.7	48.9
max, levenshtein, jaccard	23.5	82.9	53.2
max, stats10, levenshtein, jaccard	50.6	83.4	67.0
max, secondiff, levenshtein, jaccard	51.2	83.0	67.1
max, secondiff	51.4	83.1	67.3

Table 1: NIL prediction feature ablation study on the dev set (trained on the train set). *max* is the score of the best candidate, *levenshtein* and *jaccard* are textual similarities between the mention and the entity title, *secondiff* is the difference between the best and the second-best score.

We use the bi-encoder for both candidate generation (through approximate nearest neighbor) and candidate ranking [35]. In addition, the bi-encoder, behaving as a mapping function from the real-world examples to the vectors cached in the index, does not require any training step neither fine-tuning in case of KB updating (a drop in performance is expected): it is therefore well-suited for *zero-shot NEL* since, as will be seen, from the dense representation it can be inferred the entity meaning in some ways analogue to the human understanding process.

In the following experiments, we used the bi-encoder proposed as the candidate generator of BLINK [35], available pre-trained on GitHub[3]. For the Background KB we used the FAISS [16] index based on the August 2019 Wikipedia dump and made available by [35], making sure we filter out those entities we set as NIL for the experiments of section 5. Despite BLINK behave worse than RELIC [19, 20] on all co-reference resolution tests, we opted for the first one as it is natively suited for zero-shot NEL.

3.2 NIL Prediction

The NIL prediction module predicts, given the candidates provided by the linking module, whether the top-ranked candidate entity is correct or not, without modifying the ranking given by the NEL module. According to this prediction, the mention is respectively linked to the top-ranked entity or set as NIL.

Our model is based on a logistic regression whose input is composed by the linking score of the best candidate and the difference between the best and the second-best score. This configuration is the result of an ablation study (Table 1) conducted using the train set for training and evaluating on the dev set of our dataset, that is presented in the next section. The study considered, in addition to the score of the best candidate for NEL, the similarity scores derived by Levenshtein distance and Jaccard index of the mention and the best candidate title, and some statistics (mean, median, and standard deviation) of the scores of the top-k candidates (to not consider only the top-1 candidate). An additional study has been performed on the AIDA dataset [38] (in order to avoid overfitting and to obtain information about the entity type), also considering the types of both the mention and the top-ranked entity; this study highlighted that statistics are strongly dependent on the training KB thus not suited for I-NEL (that modifies the KB) and that types can give an important contribution to the NIL prediction, but unfortunately our dataset lacks typing information. The output of logistic regression belongs to $[0, 1]$ and represents the confidence of the linking between the mention and the best candidate (close to 0 means NIL). If the prediction is NIL, the mention is then passed to the NIL clustering step in order to check if there are other mentions of the same entity. The scores of the bi-encoder, on which the model in the experiments below relies, are not normalized (dot-product) and strongly depend on the embedding algorithm.

[3]https://github.com/facebookresearch/BLINK

3.3 NIL Clustering

The NIL clustering module aims to group together the mentions referring to the same entity. The process is made by steps, considering both the *surface form* and the embedding of the mentions. In this way, the model is not easily fooled by homonyms. As a result, for each cluster of mentions we can obtain a representation of the corresponding entity using, for instance, the medoid vector of the cluster.

We considered three different clustering approaches: the first, GNN_B, uses a greedy nearest neighbor clustering (GNN) algorithm on the dense vectors of the mentions obtained by the bi-encoder. GNN consists in clustering a mention m with all the other mentions whose similarity with m is higher than a predefined threshold. This algorithm combined with different vectorizers obtained promising results in [20].

The second clustering method, GNN_F, relies on GNN but with a feature-based vectorizer. We used the same feature-based vectorizer of [30], which uses character skip bigram indicator vectors to encode the surface text, and tf-idf vectors to represent contexts. The clustering thresholds for both GNN_B and GNN_F have been calculated so that the number of predicted clusters on the dev dataset approximately matches the number of unique entities [20].

The third clustering method, $3Steps$, is a three-steps algorithm [25]: initially the mentions are clustered using their surface form (edit-distance ≤ 3 for words longer than 3 characters, perfect match otherwise). Then, inside each cluster we apply a hierarchical clustering algorithm, splitting to a predefined threshold, considering the mentions' dense vectors provided by the bi-encoder that correspond to a semantic representation of the mention. In this way, each cluster from the first step is divided into sub-clusters. Finally, semantically similar sub-clusters are merged together, according to the distance between their medoid vectors.

Novel Entities Representation. New entities lack a meaningful description that is required by the bi-encoder to represent them. Taking advantage of the dense representation of the mentions, we are able to obtain a vector for a new entity even without a description. This process can be seen as to infer the meaning of an entity from real-world examples. We tested three possible solutions to represent new entities (that are clusters of NIL mentions):

- using the vector of the *first* mention encountered;
- using the *medoid* vector of the mentions of the cluster at the time in which it is stored into the KB;
- including in the KB *all* the mentions vectors, as they were different entities.

Table 2 summarizes the performance of the bi-encoder using each one of the three strategies, as well as their time and disk

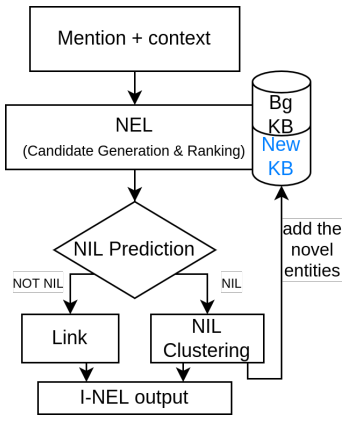

Figure 2: Schema of the pipeline.

Table 2: Recall results of the linking from scratch experiment (on AIDA [38]) with retrieval time and disk requirements.

	R_1	R_3	R_{10}	R_{30}	#vectors	time(s)	disk(MB)
first	73.7	85.6	91.9	95.8	4022	2.4	16
medoid	79.5	91.1	95.9	98.7	4022	2.4	16
all	93.9	96.7	98.5	99.3	18319	44.1	72

requirements. Tests have been conduced on AIDA for the same reasons of the NIL predictor.

Despite the latter strategy obtains the highest recall values, we decided to use the medoid vector to save resources. In fact it takes 5% of the time and it requires 22% of the disk space. This strategy indeed represents a good trade-off between efficiency and effectiveness.

Despite the human intervention, the problem of finding the better representation for new entities still persists: a validator may annotate an unrepresentative set of mentions for a given entity, since embeddings are not interpretable, introducing a bias in next batches.

4 TRANSFORMING A NEL BENCHMARK INTO AN INCREMENTAL ENTITY EXTRACTION BENCHMARK

In order to test our pipeline in a realistic scenario, the chosen test dataset should be representative of characteristics that we expect to find in the real world. For example, the entity frequency distribution should have a long right-tail: there are a few popular (high-mentioned) entities, while most are not well-known; entities in train and test data should belong to the same domains (domain adaptation is an interesting problem, but is left out of this evaluation); the dataset should be big enough to easily train data-hungry models and to have a test set that can be split into several batches.

The most valuable candidate datasets for being adapted to the incremental scenario are: AIDA [38], Zero-shot EL dataset [21], KORE50 [10], TACKBP-2010 [13], and WikilinksNED Unseen-Mentions (WNUM) [27]. In this paper we applied the following methodology to WNUM, because it is the only one freely available with the above mentioned features (e.g., AIDA is smaller than WNUM and has no ground truth on NILs) and uses links to Wikipedia entities, as well as most state-of-the-art NEL algorithms which use Wikipedia as reference KB [3, 6, 35]. Observe that even if Wikipedia may be not considered a proper KB, links between Wikipedia and Wikidata or DBpedia exists, in such a way that information about most of Wikipedia entities can be collected from proper KBs. In addition, it is designed so that each of the sets (train, dev, test) never contains a mention-entity pair that is present in another set, which is a highly appreciated property for our task.

Table 3: Statistics about the dataset before and after the *transplant*.

	mentions	(NIL)	entities	(new)
train	2.2M	(25744)	86184	(17957)
dev	10k	(316)	2397	(61)
test	10k	(307)	2514	(63)
train	2.008M	(25365)	81858	(17619)
dev	100k	(501)	7105	(214)
test	100k	(501)	6473	(248)

New entities. In order to simulate the presence of new entities, we randomly flag some entities as NILs, preserving the ground truth. This process is made in a proportional way with the number of mentions of the entity itself in the train set and so that a certain number of new entities can be chosen arbitrarily. For each entity we calculate the score:

$$p_{NIL}(x) = p^{\frac{\#x}{M}} \quad \in (0, +1] \tag{1}$$

Where p is the desired percentage of NIL entities (we set it to $p = 0.1$ [1]); M is the median of entity frequencies in the train set (the mean provides inaccurate results due to the presence of a long tail of low-frequencies entities), and $\#x$ is the number of mentions referring to the entity x in the train set. This function is demonstrated to be monotonically decreasing, providing a higher number of New Entities which are mentioned only once in the train set, since, conceptually, there are a lot of new entities which are not included due to data quality matters (low mentioned) and a few entities which may become popular (in the meaning of "high mentioned") in a small span of time.

Then, we flag each entity as new (NIL) according to a Bernulli test with probability $p = p_{NIL}$; entities flagged as NILs are removed from the BG-KB, since they become unknown. At this point, some mentions of NIL entities are *transplanted* from the train set to the dev and test sets only to increase the number of new entities in these latter sets (see Table 3), obtaining more robust metrics: this step is made randomly so that both dev and test sets have 500 new mentions each. Finally, we divide the test set in 10 batches with a stratified sampling on entity frequencies and NILs. Table 4 shows several per-batch statistics about the dataset.

5 EVALUATION

To evaluate the I-NEL we need to consider that, given a mention m that refers to an entity E, the correct behavior may depend also on previous batches: in case E is not present in the BG-KB at time t_i, m should be classified as NIL; but, if one of the previous batches

Table 4: Per-batch statistics about the test set: number of mentions, entities, NIL mentions, new entities, and $|Prev\ E|$: new entities already found in a previous batch (that should be linked to an entity previously added to the NEW-KB)

	$\|M\|$	$\|E\|$	$\|NIL\ M\|$	$\|NIL\ E\|$	$\|Prev\ E\|$
b_0	10k	2.5k	50	37	-
b_1	10k	2.5k	50	35	12
b_2	10k	2.5k	50	44	15
b_3	10k	2.5k	50	41	16
b_4	10k	2.5k	50	38	10
b_5	10k	2.5k	50	40	18
b_6	10k	2.5k	51	46	22
b_7	10k	2.5k	50	41	19
b_8	10k	2.5k	50	40	22
b_9	10k	2.5k	50	44	24
ALL	100k	6.5k	501	248	-

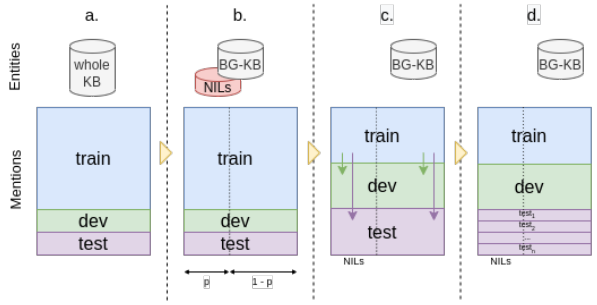

Figure 3: Construction of the I-NEL dataset from a NEL dataset (a): first $p\%$ of mentions from the corpus are flagged as NILs and corresponding entities are removed from the KB (b); then observations are just *transplanted* in order to obtain a well-represented distribution for the evaluation (c); finally the test set is split in batches (d).

contained a mention of E, the system should have already added it to the NEW-KB and therefore m should be linked to E. Figure 4 summarizes how mentions should be processed according to the entity to which they refer.

The evaluation procedure for the I-NEL task should calculate the overall performance of a given system as well as its specific performance on the subset of mentions that, in each batch, needs to be (a) linked to the BG-KB, (b) classified as NIL, or (c) linked to the NEW-KB (see Figure 4).

Finally, to better understand the impact of the error propagation between batches, the I-NEL evaluation needs to be comparable with a standard single-batch approach.

5.1 Evaluation Measures

The evaluation of an incremental pipeline is not intuitive compared to a single model, since the error propagates not only through the pipeline but also through time (in the meaning of batches).

First, we define the measures to evaluate the whole pipeline on the I-NEL task:

- (a) the "Link to BG-KB": the accuracy with the mentions that should be linked to the BG-KB;
- (b) the "NIL": the accuracy with the mentions that should be classified as NIL;
- (c) the "Link to NEW-KB": the accuracy with the mentions that should be linked to the NEW-KB;
- (d) the "Overall Accuracy" as an overall score on all the mentions.

Then, in order to understand how each component of the pipeline behaves for its specific task, we additionally evaluate each module separately:

- NEL: we calculate *Recall*@1 for the candidate generation task; since the linking is made with the first candidate. Note that this metric coincides with accuracy.
- NIL predictor: we calculate precision, recall, and f1-score of the "NIL" class.
- NIL clustering: similar to the co-reference resolution task [20], we calculate precision, recall, and f1-score of the three metrics MUC, B_3, and $CEAF_e$. We also provide the average of the f1-scores.

The clustering procedure, however, requires some further explanations for the error analysis. In particular, remember that after clustering novel entities are added to the KB: a mention which refers to an entity previously added in the NEW-KB is treated correctly if linked to that new entity, while labeling it as NIL is an error. This precaution avoids obtaining high performance in case the NIL clustering presents each mention as a novel entity.

In addition, the NIL prediction mitigates NEL errors: if the linking is wrong, the NIL predictor would ideally classify the mention as NIL because the entity offered by the linker is not correct, despite the presence of the correct entity in the BG-KB. Obviously, if the NIL predictor does not classify the mention of a novel entity as NIL, it is considered as an error.

Finally, if a new entity is erroneously created due to a false positive in the NIL prediction while the correct entity already exists in the BG-KB, each mention linked to this incorrect new entity is considered as an error.

5.2 Experiments

To allow the comparison with other models, we decided to test our baselines with the following experiments: the first one runs the evaluation on the whole test set (*one-pass*), with only one step of clustering; the second experiment, instead, is an incremental evaluation on the 10 batches of the test set. This second experiment is the focus of the present work and simulates the arrival of new documents in the pipeline. Note that the evaluation process is different in the two experiments, since the NIL clustering has side-effects: it adds novel entities to the NEW-KB that are linkable in the following batches. The first experiment has two main purposes: it provides results comparable with other models in literature and permits to estimate the drop in performance of the incremental procedure. We evaluated the three baselines (that differ in the clustering approach) using these two experimental setup, although in table 5 we report only the NIL clustering performance for all the baselines, while the

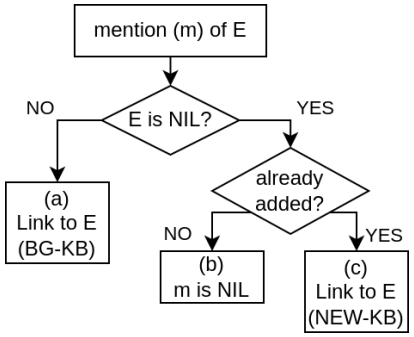

Figure 4: Schema of the expected outcome of the system, given a mention m referring to an entity E, when (a) E is known "a priori", (b) E has already been added while processing the previous batches, (c) E is not in the KB.

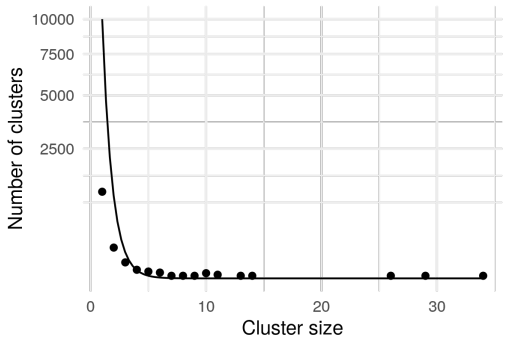

Figure 5: The absolute frequency of clusters by size (the points) vs the expected values (the curve) from the test-set.

remaining metrics are obtained using the top-performing clustering approach (3$Steps$). Finally, we run another experiment, called "correct", that corrects the output of previous components before proceeding through the batches and the pipeline, to better study the error-propagation. In able 5 we show the "correct" performance only of the top-performing pipeline (3$Steps$).

5.3 Results

In order to investigate which component of the pipeline has a stronger contribute to the final error, we analyzed each component one by one.

The results of the most relevant experiments are reported in table 5. We can see that the performance of the NEL module is competitive compared to more complex models of state of the art: in particular, GENRE [6] obtains a $R@1_{total} = 74.7$ and $R@1_{unseen} = 70.4$, which are quite similar to ours (excluding error propagation). The drop in performance, as expected, is due to the error propagation in the incremental procedure and not to a poor NEL model. Indeed, in the "correct" experiment, there is no performance drop caused by the error propagation.

The NIL prediction component, despite the high precision, yields a high number of false negatives (entities that are predicted to be in the KB while they were new). A low recall translates in a low number of false NILs, which drastically decreases the error propagation through time.

With respect to the NIL clustering, analyzing and well understanding this component is fundamental, since it has side effects that propagate through following steps, and therefore presents different results if run one batch by one or on a single pass. Among the evaluated approaches the top-performing one is the 3$Steps$; the reason could be that this method exploit a combination of lexical and semantic similarities between the mentions, while the other two (GNN_B and GNN_F) only exploit semantic dense representations.

The NIL predictor introduces two critical problems in the pipeline: false positives of the NIL predictor propagate errors through batches causing the NEW-KB to be full of redundant representations of the same entities; on the other hand, when NIL mentions are erroneously identified as not-NIL, the pipeline misses mentions useful to represent a new entity. For this reason, to mitigate error propagation through time, it is preferable to achieve higher precision than higher recall. To visualize an overview of the process, figure 5 shows that the number and size of clusters predicted by the pipeline is approximately as expected. A higher precision translates in a lower number of clusters, for the most representing novel entities; while a higher recall translates in a higher number of false representations of entities already included in the BG-KB.

Those errors could be mitigated by a HITL process, which merges clusters representing the same entities (fixing errors of the NIL predictor) or splits a cluster, removing wrong mentions.

5.4 Discussion: Challenges in Incremental Entity Extraction

The results demonstrate that error propagation across batches is a key challenge in incremental entity extraction. A drop in performance can be especially observed on NEL results, as shown in table 5: the results obtained in the first batch are comparable to the ones obtained in the one-pass experiment and with the "correct" experiment, while they deteriorate in the following batches.

As previously explained, a major source of errors came from the false positives introduced by the management of NILs, suggesting that effective NIL prediction is an important challenge for the I-NEL task; in fact, low precision leads to the introduction of error into the pipeline and, therefore, to performance deterioration.

Finally, in our experiments, we use a single vector for novel entity representation: the medoid obtained from the clusters of the NIL mentions. Evidence (see Table 2) show that this method is sub-optimal, despite being more efficient. The problem of how to represent new entities, for instance using more than one vector per cluster or with different strategies (e.g. bounding-boxes), is another important challenge in the I-NEL task that is worth studying in order to find a better compromise between effectiveness and efficiency. We leave this exploration to future works. Also the problem of deciding when to update the KB is quite interesting (the index is now updated after having processed the first batch in which the new mention appears): it may be the case that collecting more observations of a new entity is necessary to obtain a good representation, possibly considering a confidence score. This aspect was

		b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	$\overline{b}$	one-pass
NEL	R@1	72.64	62.03	55.43	51.68	46.53	45.02	41.71	40.56	39.03	38.39	49.30	72.91
NIL Pred	P	65.16	66.25	70.55	74.30	78.92	78.64	80.00	79.53	81.38	81.96	75.67	64.01
	R	46.74	30.51	30.55	31.47	33.59	34.14	34.32	34.21	35.79	36.10	34.74	46.06
	F1	54.43	41.78	42.63	44.22	47.13	47.61	48.04	47.84	49.71	50.12	47.35	53.57
NIL Clust: comparison among 3*Steps*, GNN_B, GNN_F: best results highlighted in bold													
3*Steps*	MUC F1	**96.84**	**95.97**	**95.89**	**96.49**	**96.96**	**97.10**	**96.94**	**97.48**	**97.26**	**97.48**	**96.84**	**97.38**
	B3 F1	**97.43**	**97.07**	**96.73**	**96.25**	**96.48**	**96.66**	**96.11**	**97.14**	**95.79**	**96.55**	**96.62**	**94.36**
	CEAF F1	**95.16**	**94.43**	**93.10**	**93.02**	**92.56**	**93.15**	**92.38**	**93.35**	**92.49**	**93.26**	**93.29**	**82.27**
	$\overline{F1}$	**96.48**	**95.82**	**95.24**	**95.25**	**95.33**	**95.63**	**95.14**	**95.99**	**95.18**	**95.76**	**95.58**	**91.34**
GNN_B	MUC F1	86.48	84.22	84.55	87.65	89.23	89.60	89.30	91.07	90.88	91.05	88.40	91.56
	B3 F1	91.33	90.12	89.07	90.34	90.92	90.74	90.36	91.69	91.72	91.92	90.82	86.46
	CEAF F1	84.96	83.94	80.62	81.65	80.95	81.58	80.77	82.57	81.38	81.14	81.96	64.61
	$\overline{F1}$	87.59	86.09	84.75	86.54	87.03	87.31	86.81	88.44	87.99	88.04	87.06	80.88
GNN_F	MUC F1	36.08	31.05	31.04	29.84	28.37	28.16	27.67	28.53	26.69	29.17	29.66	65.18
	B3 F1	71.66	65.16	58.68	56.80	55.03	54.78	53.31	54.40	51.89	51.16	57.29	48.83
	CEAF F1	62.72	56.91	49.09	47.01	44.90	44.60	43.24	44.22	41.66	40.46	47.48	35.89
	$\overline{F1}$	56.82	51.04	46.27	44.55	42.77	42.51	41.41	42.39	40.08	40.26	44.81	49.96
Overall	(a) Link	65.67	56.05	49.68	46.36	41.69	39.87	36.67	35.29	33.99	33.46	43.87	
	(b) NIL	42.00	38.46	51.35	32.35	60.00	50.00	41.18	46.67	44.44	37.93	44.44	
	(c) Link New	n/a*	36.36	76.92	68.75	73.33	70.00	70.59	85.00	69.57	61.90	61.24	
	(d) Acc	65.55	55.96	49.72	46.35	41.80	39.96	36.74	35.42	34.10	33.53	43.91	
Correct NEL	R@1	72.64	72.11	72.24	72.79	72.26	73.13	71.71	72.77	71.87	72.28	72.38	
NIL Pred	P	55.43	55.17	54.87	54.26	55.52	55.61	55.96	56.21	55.83	54.38	55.32	
	R	43.32	42.43	45.47	43.02	43.57	43.44	42.22	44.91	42.30	43.08	43.38	
	F1	48.63	47.97	49.73	47.99	48.83	48.78	48.13	49.93	48.13	48.08	48.62	
NIL Clust	MUC F1	100.00	80.00	-	100.00	66.67	66.67	-	100.00	-	-	51.33	
3*Steps*	B3 F1	100.00	97.26	100.00	100.00	95.65	97.78	100.00	100.00	97.14	100.00	98.78	
	CEAF F1	100.00	96.89	100.00	100.00	94.53	96.12	100.00	100.00	95.24	100.00	98.28	
	$\overline{F1}$	100.00	91.38	66.67	100.00	85.62	86.86	66.67	100.00	64.13	66.67	82.80	

Table 5: Evaluation results: *NEL*, *NIL Pred*, *Overall*, and *Correct* are obtained using the pipeline with the 3*Steps* clustering algorithm (the top performing one). *Correct* results are obtained correcting the output of the previous components. NIL prediction performance are calculated considering correct when a NEL error is mitigated. *Nothing can be linked to previously added entities in b_0. "-" represents cases where the gold standard contains one element per cluster; in this case, *MUC F1* score (= 0) is not meaningful [28].

ignored at the moment since it is highly correlated to the long-tail entities problem (in the meaning of "mentioned few times"): in fact the dataset was modified in order to add a high number of new entities mentioned only once. Finding a good trade-off between getting useful representations and handle entities mentioned only once, is an interesting problem for future work.

6 CONCLUSION

In this work, we introduce the task of incremental NEL (I-NEL), providing both a dataset and a simple pipeline, both as baseline and as a workbench to identify critical components.

Our experiments show that a main challenge for I-NEL is to deal with the incremental propagation of error, which is due, especially, to the difficulty of linking entity mentions found in one batch to novel entities, i.e., entities not in the background KB and added to the NEW-KB in previous batches. Finding representations of novel entities that provide a reasonable trade-off between efficiency and effectiveness is not trivial. Indeed, the performance of the NEL component [35] is reasonably good at the beginning but worsen over time. Another component that must be improved to deliver robust incremental NEL is NIL prediction.

Future works include the application of our methodology to generate incremental versions of more datasets. In addition, we plan to elaborate on methods to address the main challenges found in I-NEL, especially in relation to the management of novel entities (NIL prediction and NIL mention representation).

The dataset, the source code of the procedure to create it, and the code for the evaluation are openly available at https://github.com/rpo19/Incremental-Entity-Extraction/.

REFERENCES

[1] Dhruv Agarwal, Rico Angell, Nicholas Monath, and Andrew McCallum. 2021. Entity Linking and Discovery via Arborescence-based Supervised Clustering.

ArXiv abs/2109.01242 (2021).

[2] Carl F. Andersen, Drew Wicke, Kyle Tunis, Wheeler Howard, Mark J. Gerken, Dustin Carroll, Cassidy Harless, Cecilia Newell, and T. Swift. 2019. KB Construction and Hypothesis Generation Using SAMSON. In *TAC*.

[3] Edoardo Barba, Luigi Procopio, and Roberto Navigli. 2022. ExtEnD: Extractive Entity Disambiguation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

[4] Carlo Batini, Valerio Bellandi, Paolo Ceravolo, Federico Moiraghi, Matteo Palmonari, and Stefano Siccardi. 2021. Semantic Data Integration for Investigations: Lessons Learned and Open Challenges. In *2021 IEEE International Conference on Smart Data Services (SMDS)*. 173–183.

[5] Kevin Blissett and Heng Ji. 2019. Cross-lingual NIL Entity Clustering for Low-resource Languages. In *Proceedings of the Second Workshop on Computational Models of Reference, Anaphora and Coreference*. 20–25.

[6] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In *International Conference on Learning Representations*.

[7] Sourav Dutta and Gerhard Weikum. 2015. Cross-Document Co-Reference Resolution using Sample-Based Clustering with Knowledge Enrichment. *Transactions of the Association for Computational Linguistics* 3 (2015), 15–28.

[8] Nicolas Rodolfo Fauceglia, Yiu-Chang Lin, Xuezhe Ma, and Eduard H. Hovy. 2015. CMU System for Entity Discovery and Linking at TAC-KBP 2015. *Theory and Applications of Categories* (2015).

[9] Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R. Curran. 2013. Evaluating Entity Linking with Wikipedia. *Artificial Intelligence* 194 (2013), 130–150.

[10] Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2012. KORE: keyphrase overlap relatedness for entity disambiguation. *Proceedings of the 21st ACM international conference on Information and knowledge management* (2012).

[11] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Real-time Inference in Multi-sentence Tasks with Deep Pretrained Transformers. *CoRR* abs/1905.01969 (2019). arXiv:1905.01969

[12] Heng Ji and Ralph Grishman. 2011. Knowledge Base Population: Successful Approaches and Challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. 1148–1158.

[13] Heng Ji, Ralph Grishman, H.T. Dang, K. Griffit, and J. Ellis. 2010. Overview of the TAC 2010 knowledge base population track. In *TAC 2010*.

[14] Heng Ji, Joel Nothman, Ben Hachey, et al. 2014. Overview of tac-kbp2014 entity discovery and linking tasks. In *TAC 2014*. 1333–1339.

[15] Shanshan Jiang, Yihan Li, Tianyi Qin, Qian Meng, and Bin Dong. 2017. SRCB Entity Discovery and Linking (EDL) and Event Nugget Systems for TAC 2017. *Theory and Applications of Categories* (2017).

[16] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547.

[17] Manling Li, Ying Lin, Ananya Subburathinam, Spencer Whitehead, Xiaoman Pan, Di Lu, Qingyun Wang, Tongtao Zhang, Lifu Huang, Heng Ji, Alireza Zareian, Hassan Akbari, Brian Chen, Bo Wu, Emily Allaway, Shih-Fu Chang, Kathleen McKeown, Yixiang Yao, Jennifer Chen, Eric J Berquist, Kexuan Sun, Xujun Peng, Ryan Gabbard, Marjorie Freedman, Pedro A. Szekely, T. K. Satish Kumar, Arka Sadhu, Ram Nevatia, Miguel E. Rodríguez, Yifan Wang, Yang Bai, Ali Sadeghian, and Daisy Zhe Wang. 2019. GAIA at SM-KBP 2019 - A Multi-media Multi-lingual Knowledge Extraction and Hypothesis Generation System. In *TAC*.

[18] Zixuan Li, Yunqi Qiu, Fan Yang, Xiaolong Jin, Yuanzhuo Wang, Yantao Jia, Haoran Yan, Kailin Zhao, and Jialin Su. 2017. The Open Knowledge System for TAC KBP 2017. *Theory and Applications of Categories* (2017).

[19] Jeffrey Ling, Nicholas FitzGerald, Zifei Shan, Livio Baldini Soares, Thibault Févry, David Weiss, and Tom Kwiatkowski. 2020. Learning Cross-Context Entity Representations from Text.

[20] Robert L Logan IV, Andrew McCallum, Sameer Singh, and Dan Bikel. 2021. Benchmarking Scalable Methods for Streaming Cross Document Entity Coreference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 4717–4731.

[21] Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-Shot Entity Linking by Reading Entity Descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 3449–3460.

[22] Xuezhe Ma, Nicolas R. Fauceglia, Yiu-Chang Lin, and Eduard H. Hovy. 2017. CMU System for Entity Discovery and Linking at TAC-KBP 2017. In *TAC 2017*.

[23] Paul McNamee and Hoa Trang Dang. 2009. Overview of the TAC 2009 knowledge base population track. In *TAC 2009*, Vol. 17.

[24] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

[25] Sean Monahan, John Lehmann, Timothy Nyberg, Jesse Plymale, and Arnold Jung. 2011. Cross-Lingual Cross-Document Coreference with Entity Linking. *Theory and Applications of Categories*.

[26] Nafise Sadat Moosavi and Michael Strube. 2016. Which Coreference Evaluation Metric Do You Trust? A Proposal for a Link-based Entity Aware Metric. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 632–642.

[27] Yasumasa Onoe and Greg Durrett. 2020. Fine-Grained Entity Typing for Domain Independent Entity Linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 8576–8583.

[28] Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand index for coreference evaluation. *Natural Language Engineering* 17, 4 (2011), 485–510.

[29] Özge Sevgili, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. 2022. Neural entity linking: A survey of models based on deep learning. *Semantic Web* 13, 3 (2022), 527–570.

[30] Luke Shrimpton, Victor Lavrenko, and Miles Osborne. 2015. Sampling Techniques for Streaming Cross Document Coreference Resolution. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1391–1396.

[31] Manaj Srivastava, David Trupiano, David Akodes, Ilana Heintz, Hannah Provenza, Bonan Min, Jay DeYoung, Lance A. Ramshaw, and Roger Bock. 2017. Adept Automatic Knowledge Discovery System for Cold Start Knowledge Base Population. *Theory and Applications of Categories* (2017).

[32] Yongmei Tan, Di Zheng, Maolin Li, and Xiaojie Wang. 2015. BUPTTeam Participation at TAC 2015 Knowledge Base Population.

[33] Siliang Tang, Yankun Ren, Xiyuan Yang, Dan Liu, Guoping Hu, Fei Wu, and Yueting Zhuang. 2017. The ZHI-EDL System for Entity Discovery and Linking at TAC KBP 2017. *Theory and Applications of Categories* (2017).

[34] Gerhard Weikum, Xin Luna Dong, Simon Razniewski, and Fabian M. Suchanek. 2021. Machine Knowledge: Creation and Curation of Comprehensive Knowledge Bases. *Found. Trends Databases* 10, 2-4 (2021), 108–490.

[35] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable Zero-shot Entity Linking with Dense Entity Retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 6397–6407.

[36] Mingbin Xu, Nargiza Nosirova, Kelvin Jiang, Feng Wei, and Hui Jiang. 2017. FOFE-based Deep Neural Networks for Entity Discovery and Linking. *Theory and Applications of Categories* (2017).

[37] Tao Yang, Dong Du, and Feng Zhang. 2017. The TAI System for Trilingual Entity Discovery and Linking Track in TAC KBP 2017. *Theory and Applications of Categories* (2017).

[38] Mohamed Amir Yosef, Johannes Hoffart, Ilaria Bordino, Marc Spaniol, and Gerhard Weikum. 2011. AIDA: An Online Tool for Accurate Disambiguation of Named Entities in Text and Tables. *PVLDB* 4, 12 (2011), 1450–1453.

[39] Ayah Zirikly, Mona T. Diab, and Yassine Benajiba. 2015. GWU English TAC-KBP EL Diagnostic Task with Name Mention. *Theory and Applications of Categories* (2015).