EDIN: An End-to-end Benchmark and Pipeline for Unknown Entity Discovery and Indexing

Nora Kassner, Fabio Petroni, Mikhail Plekhanov, Sebastian Riedel, Nicola Cancedda
Meta AI
kassner@meta.com

Abstract

Existing work on Entity Linking mostly assumes that the reference knowledge base is complete, and therefore all mentions can be linked. In practice this is hardly ever the case, as knowledge bases are incomplete and because novel concepts arise constantly. We introduce the temporally segmented *Unknown Entity Discovery and Indexing* (EDIN) *-benchmark* where unknown entities, that is entities not part of the knowledge base and without descriptions and labeled mentions, have to be integrated into an existing entity linking system. By contrasting EDIN with zero-shot entity linking, we provide insight on the additional challenges it poses. Building on dense-retrieval based entity linking, we introduce the end-to-end EDIN-*pipeline* that detects, clusters, and indexes mentions of unknown entities in context. Experiments show that indexing a single embedding per entity unifying the information of multiple mentions works better than indexing mentions independently.

1 Introduction

Most existing works on Entity linking (EL) – the fundamental task of detecting mentions of entities in context and disambiguating them against a reference knowledge base (KB) – assume that such KB is complete, and therefore all mentions can be linked. In practice this is hardly ever the case, as KBs are incomplete when they are created and because novel concepts arise constantly. For example, English Wikipedia, often used as the reference KB for large scale linking, is growing by more than 17k entities every month.[1]

Consequently, at the time of deployment EL systems are quickly outdated and static evaluation overestimates performance. But as these systems play significant role in many real world industry applications, e.g., moderating discussions around recent events, a dynamic look on EL is crucial.

Nonetheless, related work on this problem is sparse. Available datasets (Ji et al., 2015; Derczynski et al., 2017; Nakashole et al., 2013) and models (Hoffart et al., 2014) are outdated and/or small scale and use features which are not readily available (Nakashole et al., 2013; Wu et al., 2016). Most importantly, they approach the problem only in parts. We revisit this problem in context of dense-retrieval and large-scale EL, e.g., EL relying on bi-encoder architecture that runs a nearest neighbor search between mention encoding and a large-scale index of entity encodings. To this end, we introduce EDIN-*benchmark* and EDIN-*pipeline* where *unknown entities*, that is entities with no available canonical names, descriptions and labeled mentions, have to be integrated into an existing EL model in an end-to-end fashion. To the best of our knowledge, EDIN-*pipeline* is the first end-to-end pipeline tackling this problem.

Note that this setting is strictly more demanding than zero-shot (zs) entity linking (Logeswaran et al., 2019), where a textual description of the zs entities is available at the time of training.

The EDIN-*benchmark* is temporally segmented into two parts, one preceding time t_1 and one preceding time t_2. With current approaches, an EL system created at t_1 is unable to create a dense-index entry – and therefore successfully link – unknown entities introduced after t_1. The task that we propose consists in adapting a model trained at t_1 using only an *adaptation dataset* – a set of new documents also mentioning unknown entities – and unsupervised techniques. There are therefore two parts to this task: i) Discovery, which consists in detecting mentions of unknown entities in the adaptation dataset and classifying them as unknown and ii) Indexing, consisting in mapping co-referring mentions of unknown entities to a single representation compatible with the entity index.

[1] https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia (09.05.2022)

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 8659–8673
December 7-11, 2022 ©2022 Association for Computational Linguistics

By introducing a clear-cut temporal segmentation EDIN-*benchmark* targets unknown entities which are truly novel/unseen to all parts of an EL system, specifically the pre-trained language model (PLM). Therefore, the EL system cannot rely on implicit knowledge captured by the PLM. This is, to the best of our knowledge, a setting that has not been explored before in the context of dense-retrieval based EL.

Temporal segmentation also lets us study effects of entity encoder and PLM degradation. We observe that precision drops for known entities in novel contexts which points to a large problem of PLM staleness also discussed by (Agarwal and Nenkova, 2021; Dhingra et al., 2022; Lazaridou et al., 2021).

We show that distinguishing known from unknown entities, arguably a key feature of an intelligent system, poses a major challenge to dense-retrieval based EL systems, as a model has to strike a delicate balance between relying on mention vs. context: context is crucial to distinguish unknown entities carrying the same name as known entities and to co-refer different mentions of the same unknown entities, while mentions are essential to distinguish unknown entities with different name but semantic similarity to existing ones.

On the side of indexing, inserting unknown entities into a space of known entities poses problems of interference with known entities in their close proximity. For instance, when first encountering mentions of BioNTech we want to create an index entry in proximity of other biotech companies but in a way that linking can still differentiate between them. We find that adapting the EL model to the updated index, is essential.

We experiment with different indexing methods. In particular, we contrast single mention-level indexing (FitzGerald et al., 2021) with indexing clusters of mentions. We find that unifying the information of multiple mentions into a single embedding is beneficial.

We summarize our contributions as follows: i) We introduce the EDIN-*benchmark*, a large scale end-to-end EL dataset where unknown entities need to be discovered and integrated into an existing entity index in an unsupervised fashion. ii) We propose the EDIN-*pipeline* in the form of an extension of existing dense-retrieval architectures. iii) We contrast this task with zs EL, and provide insight on the challenges it poses. iv) We show that indexing a single embedding per entity, unifying the information of multiple mentions, works better than indexing mentions independently.

Data and evaluation code is located here: https://github.com/facebookresearch/EDIN

2 Task definition

We formally define end-to-end EL as follows: Given a paragraph p and a set of known entities $E_K = \{e_i\}$, each with canonical name, the title, $t(e_i)$ and textual description $d(e_i)$, our goal is to output a list of tuples, $(e, [i, j])$, where $e \in E_K$ is the entity corresponding to the mention $m_{i,j}$ spanning from the i^{th} to j^{th} token in p. We call a system that solves this task based on $d(e_i)$ a **Description-based** entity linking system L.

For EDIN-*benchmark*, after training a model L_{t1} at time step t_1, a set of unknown entities $E_U = \{e_i\}$ with $E_U \bigcap E_K = \emptyset$ and no available canonical names, descriptions and labeled mentions is introduced between t_1 and $t_2 > t_1$. The task is to adapt L_{t1} in an unsupervised fashion such that it can successfully link mentions of $E_U \bigcup E_K$.

We use three dataset splits: the training set D_{train} to train L_{t1}, the adaptation dataset D_{adapt} used to adapt L_{t1} and the test set D_{test} to evaluate. Both D_{adapt} and D_{test} include mentions between t_1 and t_2. The model relies on D_{adapt} to discover E_U and extract representations to integrate E_U into the entity index. We ensure that D_{adapt} and D_{test} are disjoint to prevent leakage of test mentions into entity representations extracted from D_{adapt}.

3 EDIN-*pipeline*

Our EDIN-*pipeline* is built on top an end-to-end extension of the dense-retrieval based model BLINK (Ledell Wu, 2020) and is similar to (Li et al., 2020). It is composed of a Mention Detection (MD), Entity Disambiguation (ED) and Rejection (R) components. MD detects entity mention spans $[i, j]$ in context relying on BERT (Devlin et al., 2019). ED links these mentions to $e \in E_K$. It relies on bi-encoder architecture running a k-nearest-neighbor (kNN) search between *mention encoding* and candidate *entity encodings* (the entity index). Mention encodings are pooled from BERT-encoded paragraph tokens $p_{1..n}$:

$$\mathbf{m}_{i,j} = FFL(BERT([CLS]p_1 \dots p_n[SEP])_{i \dots j})$$

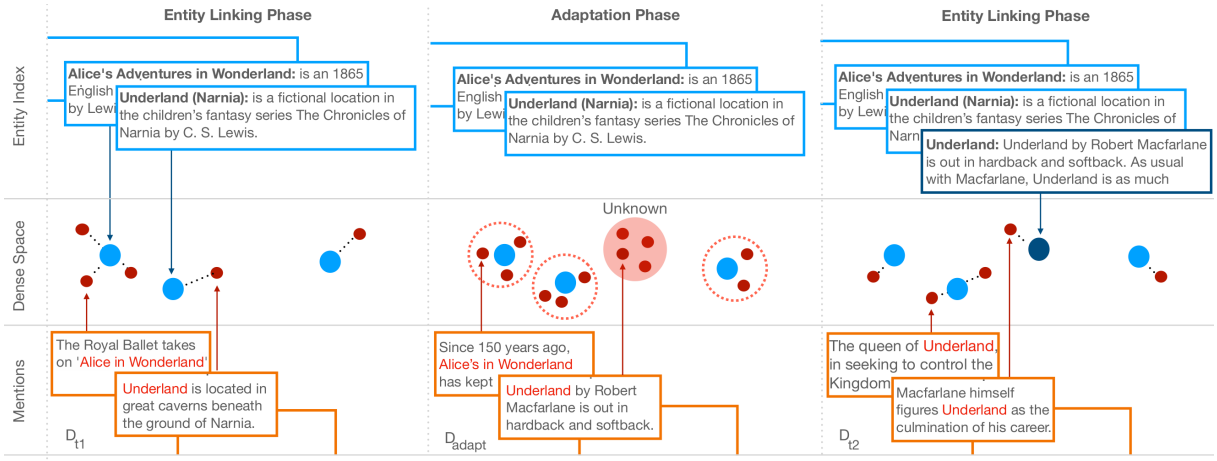

Figure 1: **EDIN-*pipeline*:** In the adaptation phase, detected mentions in D_{adapt} are mapped into a joint dense space with E_K representations. A clustering algorithm groups mentions and entities based on kNN-similarity. Clusters of mentions without entity encoding are collected in E'_U. To integrate these into the index of E_K, mentions in single-sentence contexts are concatenated and mapped to a single embedding using the entity encoder. After adaptation, the updated entity index is used for standard EL in an inductive setting.

Entities are represented using BLINK's *frozen* entity encoder:

$$\mathbf{e} = BERT_{[CLS]}([CLS]t(e)[SEP]d(e)[SEP])$$

Mention-entity candidates are passed to R that controls precision-recall trade-off by thresholding a learned candidate score.

More information about architecture and training are detailed in appendix A.

4 Unknown Entity Discovery and Indexing

We introduce an end-to-end pipeline to encode E_U into L_{t1}'s entity index. The process is depicted in Figure 1. This pipeline is fully unsupervised and only relies on D_{adapt}. It follows a two-step process: i) in Discovery the EL system detects mentions of unknown entities and recognises them as being unknown; ii) during Indexing, co-referring mentions of unknown entities are mapped to a single embedding compatible with the entity index. After adaptation the updated model is tested on D_{test}.

4.1 Unknown Entity Discovery

First, L_{t1} detects and encodes mentions part of D_{adapt}. The MD head is trained to detect mentions leveraging the context around them, and can therefore detect mentions of both E_K and E_U. Encoded mentions $\mathbf{M} = \{\mathbf{m}_1, ..., \mathbf{m}_{|M|}\}$ are then input to a clustering algorithm that partitions M into disjoint clusters $C = \{c_1, ..., c_{|C|}\}$. We adopt the same greedy NN clustering algorithm as Logan IV et al. (2021) where $\mathbf{m}_i$ is assigned to cluster c_k if $\mathbf{m}_j \in c_k$ is NN mention to $\mathbf{m}_i$ and $sim(\mathbf{m}_i, \mathbf{m}_j) > \delta$.

Next, entity encodings of $e \in E_K$ are assigned to these clusters if $\sum_{j=0..J}(sim(\mathbf{e}_i, \mathbf{m}_j)))/J > \tau$ holds for $\mathbf{m}_j \in c_i$ with $\mathbf{e}_i$ being the nearest entity of $\mathbf{m}_j \in c_i$. δ and τ are tuned on D_{adapt}-dev to optimize for recall. For more details see appendix C. Following Agarwal et al. (2021), all clusters not containing any entity representation are deemed to refer to entities in E_U. We refer to this subset of automatically identified unknown entities as E'_U.

4.2 Unknown Entity Indexing

Next, clusters identified as E'_U are integrated into the EL index of L_{t1}. We explore two different methods of indexing:

Cluster-based: We concatenate all mentions part of a cluster, each with the sentence they occur in, and use the entity encoder to map to a single entity encodings. We pool over all $m_i \in c_i$ and select the most occurring mention as canonical name $t(e)$.

Mention-based: Mentions in single sentence contexts are indexed individually using the entity encoder. Individual mentions are used as $t(e)$.

5 Evaluation

As mentions of type E_U are significantly less frequent than mentions of type E_K, we report results on these two types separately.

For *Discovery*, we report precision and recall of E_U classification and clustering metrics.

	Wikipedia	OSCAR
Train	100k (908k)	100k (1.7M)
Adapt	17k (183k)	17k (380k)
Dev Train	8k (78k)	8k (142k)
Dev Adapt	-	9k (183k)
Test	198k (1.8M)	569k (11M)

Table 1: **Dataset Statistics:** Number of samples (number of mentions) for training, adaptation and testing.

Bin	Support	E_K R	Support	E_U R
[0)	68,241	21.1	7,095	17.5
[1)	59,227	29.1	3,923	25.9
[1, 10)	313,232	45.6	9,939	40.7
[10, 100)	901,857	65.7	7,765	57.3
[100, 1k)	2,860,880	76.9	7,399	64.4
[1k, +)	5,981,028	84.4	6,717	86.7

Table 2: **Frequency effects:** End-to-end EL performance of upper baseline model L_{t2} per frequency bins.

To evaluate end-to-end EL, we compute precision (P) and recall (R) following Li et al. (2020) but using a hard matching criteria.

To do so for cluster-based discovery, canonical names of indexed clusters need to be consistent with the set of test labels. Our method of assigning canonical names to clusters based on pooling over mentions is not. To resolve this mismatch we pool over the gold labels associated with these mentions instead of the mentions themselves. This is only done for evaluation.

Unsupervised clustering of mentions in D_{adapt} may suffer from two kinds of errors: i) Clusters can be incomplete, e.g., mentions of a single entity can be split into multiple clusters which can lead to indexing the same entity multiple times and ii) Clusters can be impure, e.g., mentions of different entities end in the same cluster, which leads to conflation of multiple entities into one representation.

In our evaluation we use the gold labels for computing standard EL metrics by associating possibly more than one cluster to each E_U, and consider a prediction correct if a mention is linked to any of the clusters associated with the correct entity. EL metrics could fail to capture shortcomings in establishing co-references between mentions though, therefore we report clustering metrics alongside EL metrics. We follow Agarwal et al. (2021) and report normalized mutual information (NMI).

6 EDIN-*benchmark*

To construct the entity index, we download Wikipedia dumps from $t1$ and $t2$ and extract entity titles and descriptions. Setting t_1 to September 2019 (the date when BLINK was trained) the KB consists of 5.9M entities, setting t_2 to March 2022 an additional set of 0.7M entities is introduced.

Wikipedia and Oscar data is created as follows.

Wikipedia: Since usually only the first mention of an entity inside a Wikipedia article is hyperlinked, we annotate a subset of Wikipedia. We use a version of L that was trained at $t2$ on a labelled non-public dataset. While noisy, these predictions are significantly better than what our best discovery and indexing methods can achieve, therefore we adopt them as pseudo-labels for the purpose of comparing approaches. As discovery and indexing methods improve, manual labelling of the evaluation data will afford more accurate measures. Wikipedia provides time stamps which enables us to separate two time splits.

OSCAR news: This dataset is based on the common-crawl dataset OSCAR (Abadji et al., 2021). We select a subset of English language news pages which we label automatically as described above. The dataset consists of 797k samples, which we split based on their publication date. We publish this dataset using stand-off annotations and code to download the relevant raw data. To enable evaluation of future versions of PLMs and EL systems, we also publish our data processing scripts.

For both types of datasets we publish two time splits: D_1, containing samples preceding $t1$, which is used to train model L_{t1} and D_2, with samples preceding $t2$, which is used to train an upper bound model L_{t2}. To adapt L_{t1}, we hold out a subset of data from between $t1$ and $t2$ to construct D_{adapt} ($D_{adapt} \cap D_2 = \emptyset$). Remaining samples are randomly split into train, dev, test. Figure 2 illustrates the different data splits. Overall dataset statistics are listed in Table 1.

To construct D_{adapt}, we follow Agarwal et al. (2021), and set the ratio of mentions of type E_U to E_K to 0.1. [2] As D_{t2}-test covers both known and unknown entities, we use this dataset for EDIN-*pipeline* evaluation. In Oscar D_{t2}-test, the average number of mentions per E_U is 5.6 and it is ten

[2]Naturally this ratio would lie at 0.02. We made this artificial adjustment to reduce the strong class imbalance and obtain more interpretable and statistically stable results. Such adjustment could be lifted once considerably more precise unknown entity discovery components become available.

times lower than for E_K. COVID-19 is the most occurring unknown entity with 12k mentions. 638k E_U are not mentioned at all and only 733 are mentioned more than ten times.

7 Results and Discussion

In the following sections, we discuss results for OSCAR data. Results on Wikipedia data are consistent but lower and shown in appendix F. Our main findings are shown in Table 3 where we report end-to-end performance on OSCAR D_{t2}-test.

Overall, our results show:

- EDIN-*benchmark* is challenging. Particularly attributed to imperfect discovery, end-to-end performance in terms of recall lacks significantly behind the upper bound, see 7.5.
- When contrasting with zs EL, we find that i) adapting the model to the updated index by re-training the model after indexing is crucial, see 7.2 and ii) entity encodings relying on clusters of mentions in context instead of human crafted descriptions have high potential but discovering these clusters is challenging, see 7.4.1.
- Our best performing system relies on Cluster-based indexing, with the advantage of attending to and unifying the information of multiple mentions, see 7.4. We call this version the EDIN-*pipeline*.

In whats to come, we first discuss upper and lower performance bounds. Then, we follow our two-step pipeline where we first present results on discovery and indexing separately and then assemble the full end-to-end pipeline.

Recall our terminology:

- **Cluster-based:** E_U encodings rely on mentions in context which are concatenated and embedded into a single encoding.
- **Mention-based:** E_U encodings rely on individually indexed mentions in context.
- **Description-based:** E_U encodings rely on human crafted descriptions. This type of indexing is used in the zs setting.

7.1 Lower and upper bounds

Our starting point, and an obvious lower performance bound, is given by model L_{t1} trained at D_{t1}. This model lacks representations of E_U and its training data does not contain any corresponding mentions. Therefore, performance on the subset of E_U is 0 for all metrics.

For an upper performance bound we take model L_{t2} trained at D_{t2}. The entities in E_U were introduced to Wikipedia past t_1 but before t_2, meaning that to L_{t2} these entities are actually *known*: labeled mentions of E_U are part of the training data and entity representations are part of the index.

L_{t2} reaches similar performance as L_{t1} for E_K. We suspect performance differences can be attributed to the difference in training data.

Performance of L_{t2} on mentions of E_U is lower than on mentions of E_K. The performance discrepancy between E_U and E_K is largely due to frequency differences, see Table 2. We suspect that the remaining difference can be attributed to the degradation of PLM and entity encoder. Note that while labelled mentions of E_U were seen during the training phase of L_{t2}, BLINK's entity encoder was not re-trained. To investigate this hypothesis further, we test L_{t1} on mentions of E_K that meet two conditions: i) time stamps of these samples are posterior to $t1$ and ii) two or more mentions of E_U occur in their context. Thus, we target mentions of E_K in novel contexts to which neither BLINK nor the PLM have been exposed. The total number of entities that meet these conditions are 40,055. We find that recall drops only slightly from 80.1 to 79.9 but precision drops from 82.0 to 75.9. This result indicates that E_U are also a source of noise when trying to link mentions of E_K.

7.2 Additional upper bound: Zero-shot EL

Zs EL relies on Description-based indexing. It may be a valid option in some practical settings, where we may e.g. be able to frequently download fresh Wikipedia snapshots and rerun all or part of the training, but it does not meet the conditions for being a valid entry to EDIN-*benchmark*, because it relies on supervision for deciding what novel entities to add to the index, and because it requires manually written descriptions for such entities. For these reasons, we present it here as an additional upper bound comparison point.

We note that in the zs problem, all entities are part of the index at training time. In the setting

	Known Entities			Unknown Entities			Unknown Entities filtered		
Model	**R**	**P**	**NMI**	**R**	**P**	**NMI**	**R**	**P**	**NMI**
L_{t1} (lower bound)	80.1	82.0	93.5	0.0	0.0	0.0	0.0	0.0	0.0
L_{t2} (upper bound)	78.7	79.7	93.1	49.2	31.8	93.8	63.1	26.0	93.4
L_{t1}-Descp (zs setting)	80.2	82.6	93.5	46.5	32.4	93.8	58.3	26.2	90.5
L_{t1}-Mention-Oracle	80.6	81.5	93.3	24.0	46.6	87.0	40.7	46.6	87.0
L_{t1}-Mention	80.3	81.9	93.4	20.5	43.7	87.6	34.5	43.5	88.7
L_{t1}-Cluster-Oracle	80.3	82.0	94.2	30.5	51.8	85.9	51.8	51.8	85.9
EDIN-*pipeline* (L_{t1}-Cluster)	80.3	81.9	93.4	20.8	43.1	85.9	35.4	43.1	85.3

Table 3: **EL performance** on OSCAR D_{t2}-test for unknown entities E_U and known entities E_K. **Left** shows end-to-end performance; **Right** shows filtered performance where mentions of E_U not part of D_{adapt} are dropped from test. **Upper/Lower bounds:** L_{t1}, trained at $t1$, uses Description-based entity encodings and constitutes the lower bound. It lacks encodings of E_U. L_{t2}, trained at $t2$, uses Description-based entity encodings and constitutes the upper bound. E_U are part of the index and their labeled mentions are part of training. L_{t1}-Descp adapts L_{t1} by adding Description-based entity encodings of E_U to the index. As it relies on human discovery and descriptions it constitutes an additional upper bound. **Adaptation:** For L_{t1}-Mention Mention-based encodings of i) oracle E_U and ii) discovered E'_U part of D_{adapt} are added to L_{t1}'s entity index. For L_{t1}-Cluster Cluster-based encodings of i) oracle E_U and discovered E'_U part of D_{adapt} are added to L_{t1}'s entity index.

of EDIN-*benchmark*, indexing happens after training. We run the following experiments to study the effect this difference has:

- **Not Re-trained:** Description-based entity representations are added to the index *without* re-training L_{t1} after indexing.
- **Re-trained:** Description-based entity representations are added to the index *with* re-training L_{t1} after indexing.

Recall and precision of E_U with *Re-trained* is 46.5% and 32.4% respectively, see Table 4. Recall and precision with *Not Re-trained* is 26% and 17% points lower respectively.

We note that unknown entities can potentially be placed in close proximity to known ones in embedding space. When these entity encodings are present during training, they can be picked up as hard negatives and the mention encoder can learn to circumvent them. This hypothesis is supported by experiments showing that the mean similarity between mentions and correct known entity embeddings increases significantly when the mention encoder is re-trained after adding the new entities. For details see the appendix D.

The take-away for the EDIN-*pipeline* is that, after adding new entity representations to the index, another round of training is needed to adapt the mention encoder to the updated index. We adopt this approach for the following experiments. Besides adapting the mention encoder, re-training BLINK could have a similar effect: in such case learning from hard negative can affect the spacing of entity encodings. As re-training BLINK is expensive, we did not explore this option.

	Unknown Entities			Known Entities		
	R	**P**	**NMI**	**R**	**P**	**NMI**
Not re-trained	20.6	15.5	95.2	80.1	82.3	93.5
Re-trained	46.5	32.4	93.8	80.2	82.6	93.5

Table 4: **Adapting the model to the updated index:** End-to-end EL performance on OSCAR D_{t2}-test when adding Description-based representation of unknown entities E_U to the entity index with (Re-trained) and without (Not re-trained) re-training of L_{t1}.

7.3 EDIN Discovery

First condition for effective discovery is the ability to reliably detect mentions of both E_K and E_U. Recall of L_{t1} on D_{adapt} for MD task is 90% for E_K and 86% for E_U. As expected, recall of mentions of E_K is higher as no mentions of E_U were seen during training. As a reference, running L_{t2} on D_{adapt} we find that for both E_K and E_U 91% of mentions are recalled. Note again, that for L_{t2}, E_U are *known*. This indicates that MD is not affected by frequency differences and PLM degradation.

Once mentions are detected, we adopt a clustering approach to classify between mentions of E_U

and E_K. We measure clustering quality of 91.2% NMI on D_{adapt}. We evaluate discovery based on these clusters by evaluating whether a discovered cluster is indeed referring to an E_U. Note, that here duplicated discovery of the same entity is not penalized. We set the minimum number of mentions per cluster to 3 and report low discovery precision (10%) but relatively high recall (86%). Overall, this results in detecting 71% of all unknown entities part of D_{adapt}.

We find that the constraint requiring that most mentions in a cluster are within a region controlled by hyper-parameter τ, as described in 4.1, is crucial. In an ablation study we drop this condition and greedily assign E_K to clusters if $sim(\mathbf{e}_i, \mathbf{m}_i) > \tau$ holds for *any* $m_i \in c_i$. This setting is similar to Agarwal et al. (2021) where a single entity-mention link is sufficient for cluster assignment. Discovery dropped to 49% recall and 8% precision.

A qualitative error analysis reveals that false negatives are mostly caused by the problem that mention embeddings of E_U (e.g. BioNTech) can have high similarity with entity embeddings of E_K (e.g. of other biotechnology companies). We suspect that this problem is particularly pronounced in our setting because EDIN-*benchmark* is large scale (up to 6 times more entities in the reference KB and up to 36 times more mentions in the clustering set compared to Agarwal et al. (2021)) with many tail entities.

Conversely, false positives are mostly due to known entities being misclassified unknown when occurring in novel contexts, e.g., "blood tests" or "vaccine" in context of COVID form distinct clusters. But, low precision in discovery is less problematic than low recall as re-training after indexing gives the ability to learn to ignore clusters of E_K.

7.4 EDIN indexing

After discovery, we need mention clusters of E_U to be integrated into the entity index.

We compare Mention-based and Cluster-based indexing. To isolate discovery and indexing performance, we first evaluate indexing using oracle clusters, where we replace the discovery method run on D_{adapt} with an oracle where mentions of E_U are discovered and clustered perfectly. Mention-based indexing performs worse than Cluster-based indexing with a gap of around 5% points, see Table 3 (left), L_{t1}-Mention-Oracle vs. L_{t1}-Cluster-Oracle. When reducing the test set to mentions of entities that were actually discoverable, the difference in recall becomes even more pronounced: 41% for Mention-based vs. 52% for Cluster-based indexing, see Table 3 (right).

Interestingly, this means that the ability to attend over multiple mentions in context and unify their information into a single embedding leads to superior representations. Note that here the entity encoder was neither trained to deal with the style of individual mentions in context nor with clusters of mentions in context. For future work, it would be interesting to see if Cluster-based indexing can be generally beneficial to EL, outside of the context of EDIN-*pipeline*.

7.4.1 Cluster-based vs. Description-based

As an upper baseline, we compare Cluster-based indexing with the zs setting which uses Description-based indexing. Zs EL does not rely on D_{adapt} but on a human's decision to add an entry to the index and therefore discovery is perfect. To isolate indexing from discovery, we again filter the test set to actually discoverable entities and assume perfect oracle clusters.

In this setting, see Table 3 (right), we find that Cluster-based-Oracle indexing performs 7% points lower than Description-based indexing in recall but 26% better in terms of precision.

The take-away is that when discovery is perfect, Cluster-based indexing relying on concatenated mentions in context instead of manually crafted descriptions has high potential. In the end-to-end setting, we see that assembling these perfect clusters is challenging.

We also want to emphasise that results in Table 3 show that EL performance on E_K is not affected by this adaptation process. Recall and precision remain, with 80.3 and 81.9, stable. We also test if this finding also holds on standard EL datasets. We compare performance on AIDA test before and after adaptation and report no difference in performance on E_K.

7.5 End-to-end pipeline

We assemble the full end-to-end pipeline. We replace oracle clusters of E_U by discovered clusters of E'_U. Errors in discovery that affect indexing are: i) misclassification of clusters as either known or unknown and ii) incomplete and impure clusters. We find that performance of Mention-based and Cluster-based indexing in terms of recall and precision converges and is significantly lower than their

oracle counterparts.

When reducing the test set to mentions of entities that were discoverable, thus part of D_{adapt}, Cluster-based indexing is 1% point better in terms of recall and 0.4 % worse in precision, Table 3 (right). When reducing the test set further to mentions of entities that were in fact discovered, recall of Cluster-based indexing is, with 58.4%, better than that of Mention-based indexing (55.5%).

We also report performance of ED with oracle mention detection in Table 6 in the appendix E. Here, we find that Cluster-based indexing is performing better than Mention-based indexing across all metrics.

We conclude that Cluster-based indexing performs better than Mention-based indexing. We call this version the EDIN-*pipeline*.

Besides yielding an index that scales in memory with the number of entities rather than the number of mentions – a significant advantage when the number of entities is already large and in view of a streaming extension – Cluster-based indexing generates fixed-size entity embeddings as a by-product that can have applications of their own and can be used to enhance PLMs (e.g., Peters et al. (2019)).

Overall, EDIN-*pipeline* performance shows that EDIN-*benchmark* is challenging. In terms of recall, end-to-end performance lacks 26% points behind the upper bound L_{t2}. In this setting, errors in discovery propagate. Most notably, we see this manifest when i) comparing Table 3 unfiltered and filtered where the recall problem of E_U becomes apparent and ii) comparing performance of oracle and automatic clusters where precision drops by 10% points.

In future work, we want to explore a setting where E_U are discovered in a streaming fashion, thus scaling up D_{adapt} and dropping the artificially imposed ratio of E_K vs. E_U. This would pose challenges in terms of scale and precision in discovery. Here, a human in the loop approach, as proposed by Hoffart et al. (2016) in the context of keeping KBs fresh, to introduce a component of supervision, might be needed.

8 Related work

EL is an extensively studied task. Prior to the introduction of PLMs, EL systems used frequency and typing information, alias tables, TF-IDF-based methods and neural networks to model context, mention and entity (Cucerzan, 2007; Bunescu and Paşca, 2006; Milne and Witten, 2008; He et al., 2013; Sun et al., 2015a; Lazic et al., 2015; Raiman and Raiman, 2018; Kolitsas et al., 2018; Gupta et al., 2017; Ganea and Hofmann, 2017; Khalife and Vazirgiannis, 2018; Onoe and Durrett, 2019).

Gillick et al. (2019) present a PLM-based dual encoder architecture that encodes mentions and entities in the same dense vector space and performs EL via kNN search. Logeswaran et al. (2019) proposed the zs EL task and show that domain adaptive training can address the domain shift problem. Subsequently, Wu et al. (2020) showed that pre-trained zs architectures are both highly accurate and computationally efficient at scale. None of these works tackle the problem of unknown entities.

Recently, FitzGerald et al. (2021) model EL entirely as mappings between mentions, where inference involves a NN search against all known mentions of all entities in the training set. In this setting mentions need to be labeled. They do not explore their approach in the setting of unknown entities.

Prior to dense retrieval-based EL, unknown entity discovery work includes: Ratinov et al. (2011) train a classifier to determine whether the top ranked EL candidate is unknown relying on local context, global Wikipedia coherence, and additional manually crafted features. Nakashole et al. (2013) introduce a model for unknown entity discovery and typing leveraging incompatibilities and correlations among entity types. Hoffart et al. (2014); Wu et al. (2016) study a variety of features for unknown entity discovery: Hoffart et al. (2014) use perturbation-based confidence measures and key-phrase representations and Wu et al. (2016) explore different feature spaces, e.g., topical and search engine features. These features are not readily available and incorporating them into PLM-based approaches is not straightforward; Ji et al. (2015); Derczynski et al. (2017) introduce shared tasks for discovery. These tasks are defined on comparatively small datasets and target only named entities; Akasaki et al. (2019) introduces a time sensitive method of discovering emerging entities relying on Twitter data.

None of these works consider unknown entities in an end-to-end setting including mention detection, unknown entity discovery and indexing. Also, we cannot use their datasets to evaluate as these entities were part of training the PLM.

In the context of named entity tagging, Mota and

Grishman (2009) showed that entity taggers can be effectively updated by incorporating contemporary unlabeled data using semi-supervised learning.

Closely related to EL is the task of cross document entity co-reference (CDC), where no reference KB is present (Bagga and Baldwin, 1998; Gooi and Allan, 2004; Singh et al., 2011; Dutta and Weikum, 2015; Barhom et al., 2019; Cattan et al., 2021a; Caciularu et al., 2021; Cattan et al., 2021b). Most recently, Logan IV et al. (2021) benchmark methods for streaming CDC, where mentions are disambiguated in a scalable manner via incremental clustering. Our work can be seen as bridging between the world of CDC and EL.

Most recently, Angell et al. (2021) introduce a new EL method using document-level supervised graph-based clustering. Agarwal et al. (2021) extend this work to cross-document EL and entity discovery. In this work, we adopt a more standard bi-encoder architecture (i.e. BLINK), with better EL scalability potential (memory linear in the number of entities and not in the number of mentions) and an existing end-to-end extension. We use a modified version of their discovery method.

9 Conclusion

This work introduced EDIN-*benchmark* and EDIN-*pipeline*. EDIN-*benchmark* is a large-scale, end-to-end EL benchmark with a clear cut temporal segmentation for *Unknown Entity Discovery and Indexing*. EDIN-*pipeline* detects and clusters mentions of unknown entities in context. These clusters of unknown mentions are then collapsed into single embeddings and integrated into the entity index of the original EL system.

Limitations

The main limitations of EDIN-*benchmark* are: i) The dataset is not human-annotated. Instead we used an upper-bound model to label data automatically. ii) We limit D_{adapt} in size and artificially adjust class imbalance between mentions of type E_U to E_K. The limited size of D_{adapt} in turn limits the discoverability of unknown entities, specifically low-frequency ones. Once progress is made in the accuracy and scalability of entity discovery, EDIN-*benchmark* can be modified to a truly dynamic setting where unknown entities are continuously discovered in a stream of incoming documents and integrated into the EL system.

EDIN-*pipeline* is tailored to dense-retrieval based EL and adapting it to different EL approaches, e.g., to generative EL systems De Cao et al. (2021), is not straightforward.

We study EDIN-*benchmark* and -*pipeline* in a monolingual setting using English language only. EDIN-*benchmark*'s extension to a multilingual setting is straight forward. OSCAR and Wikipedia data are available in 166 different languages but coverage will be a problem. EDIN-*pipeline* can be extended to more languages by following (Botha et al., 2020) but EDIN performance is expected to vary across languages as it does for standard EL.

EDIN-*benchmark* covers news and Wikipedia domain entities only, and we have not evaluated the EDIN-*pipeline* on other domains.

The overall performance of EDIN-*pipeline* has ample margins for improvement, with the precision of clustering-based discovery as the main bottleneck at present. The significant number of false positives (mentions of known entities classified as unknown) is still a barrier to deployment in most real-world settings.

Ethical Considerations

EL is a standard NLP task. Outside of academia EL can be deployed in both non-problematic (e.g., content understanding for hate speech detection) and problematic (e.g., surveillance) settings. Independent of the use-case, potential bias that these models could exhibit needs to be evaluated. EL relies on human curated knowledge bases (here Wikipedia) which could carry bias e.g. in terms of language, genders and races, see for example Sun and Peng (2021). Another source of bias in the context of dense-retrieval based EL, is the bias of the underlying language model (here BERT). Both potential sources of bias could be propagated to the down-stream task. To mitigate biases, we refer to Goldfarb-Tarrant et al. (2021); Steed et al. (2022) that show bias mitigation needs to be done on the side of the downstream task rather than the language model. Rudinger et al. (2018); Zhao et al. (2018) introduce methods of downstream bias mitigation, here in the context of co-reference resolution.

We publish our dataset/scripts that generate the datasets. Our dataset is based on English Wikipedia and a subset of English online news pages extracted from OSCAR. All Wikipedia based data is made fully available. OSCAR is common-crawl based data and only available to researchers upon

request. We release code and stand-off annotations which enables researchers to reproduce the dataset. Our EL annotations rely on an upper bound model which is due to the performance gap sufficient for EDIN but should not be considered gold data for general EL tasks. We will indicate this prominently on the website we use to host the data.

Acknowledgements

We thank our colleagues Louis Martin and Frederic Dreyer for their valuable feedback.

References

Julien Abadji, Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2021. Ungoliant: An optimized pipeline for the generation of a very large-scale multilingual web corpus. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-9) 2021. Limerick, 12 July 2021 (Online-Event), pages 1 – 9, Mannheim. Leibniz-Institut für Deutsche Sprache.

Dhruv Agarwal, Rico Angell, Nicholas Monath, and Andrew McCallum. 2021. Entity linking and discovery via arborescence-based supervised clustering.

Oshin Agarwal and Ani Nenkova. 2021. Temporal effects on pre-trained models for language processing tasks.

Satoshi Akasaki, Naoki Yoshinaga, and Masashi Toyoda. 2019. Early discovery of emerging entities in microblogs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 4882–4889. ijcai.org.

Rico Angell, Nicholas Monath, Sunil Mohan, Nishant Yadav, and Andrew McCallum. 2021. Clustering-based inference for biomedical entity linking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2598–2608, Online. Association for Computational Linguistics.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98/COLING '98, page 79–85, USA. Association for Computational Linguistics.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189, Florence, Italy. Association for Computational Linguistics.

Jan A. Botha, Zifei Shan, and Daniel Gillick. 2020. Entity Linking in 100 Languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7833–7845, Online. Association for Computational Linguistics.

Razvan Bunescu and Marius Paşca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 9–16, Trento, Italy. Association for Computational Linguistics.

Avi Caciularu, Arman Cohan, Iz Beltagy, Matthew Peters, Arie Cattan, and Ido Dagan. 2021. CDLM: Cross-document language modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2648–2662, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yixin Cao, Lei Hou, Juan-Zi Li, and Zhiyuan Liu. 2018. Neural collective entity linking. In *COLING*.

Arie Cattan, Alon Eirew, Gabriel Stanovsky, Mandar Joshi, and Ido Dagan. 2021a. Realistic evaluation principles for cross-document coreference resolution. In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 143–151, Online. Association for Computational Linguistics.

Arie Cattan, Sophie Johnson, Daniel S. Weld, Ido Dagan, Iz Beltagy, Doug Downey, and Tom Hope. 2021b. Scico: Hierarchical cross-document coreference for scientific concepts. In *3rd Conference on Automated Knowledge Base Construction*.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 708–716, Prague, Czech Republic. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *International Conference on Learning Representations*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Jeremy Cole, Julian Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10(0):257–273.

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*, 3:15–28.

Nicholas FitzGerald, Dan Bikel, Jan Botha, Daniel Gillick, Tom Kwiatkowski, and Andrew McCallum. 2021. MOLEMAN: Mention-only linking of entities with a mention annotation network. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 278–285, Online. Association for Computational Linguistics.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *ACL*.

Chung Heong Gooi and James Allan. 2004. Cross-document coreference on a large scale corpus. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 9–16, Boston, Massachusetts, USA. Association for Computational Linguistics.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2681–2690, Copenhagen, Denmark. Association for Computational Linguistics.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013. Learning entity representation for entity disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 30–34, Sofia, Bulgaria. Association for Computational Linguistics.

Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. 2014. Discovering emerging entities with ambiguous names. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, page 385–396, New York, NY, USA. Association for Computing Machinery.

Johannes Hoffart, Dragan Milchevski, Gerhard Weikum, Avishek Anand, and Jaspreet Singh. 2016. The knowledge awakens: Keeping knowledge bases fresh with emerging entities. *Proceedings of the 25th International Conference Companion on World Wide Web*.

Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of TAC-KBP2015 tri-lingual entity discovery and linking. In *Proceedings of the 2015 Text Analysis Conference, TAC 2015, Gaithersburg, Maryland, USA, November 16-17, 2015, 2015*. NIST.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Sammy Khalife and Michalis Vazirgiannis. 2018. Scalable graph-based individual named entity identification. *CoRR*, abs/1811.10547.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, Brussels, Belgium. Association for Computational Linguistics.

Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: Assessing temporal generalization in neural language models. In *Advances in Neural Information Processing Systems*, volume 34, pages 29348–29363. Curran Associates, Inc.

Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A selective context model for entity resolution. *Transactions of the Association for Computational Linguistics*, 3:503–515.

Martin Josifoski Sebastian Riedel Luke Zettlemoyer Ledell Wu, Fabio Petroni. 2020. Zero-shot entity linking with dense entity retrieval. In *EMNLP*.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient one-pass end-to-end entity linking for questions. In *EMNLP*.

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)*, pages 4717–4731, Online. Association for Computational Linguistics.

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*, pages 3449–3460, Florence, Italy. Association for Computational Linguistics.

David N. Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *CIKM '08*.

Cristina Mota and Ralph Grishman. 2009. Updating a name tagger using contemporary unlabeled data. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 353–356, Suntec, Singapore. Association for Computational Linguistics.

Ndapandula Nakashole, Tomasz Tylenda, and Gerhard Weikum. 2013. Fine-grained semantic typing of emerging entities. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1488–1497, Sofia, Bulgaria. Association for Computational Linguistics.

Yasumasa Onoe and Greg Durrett. 2019. Fine-grained entity typing for domain independent entity linking. *CoRR*, abs/1909.05780.

Yasumasa Onoe and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking. In *AAAI*.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *AAAI*.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1375–1384, Portland, Oregon, USA. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *NAACL*.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 793–803, Portland, Oregon, USA. Association for Computational Linguistics.

Ryan Steed, Swetasudha Panda, Ari Kobren, and Michael L. Wick. 2022. Upstream mitigation is not all you need: Testing the bias transfer hypothesis in pre-trained language models. In *ACL*.

Jiao Sun and Nanyun Peng. 2021. Men are elected, women are married: Events gender bias on wikipedia. *ArXiv*, abs/2106.01601.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015a. Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015b. Modeling mention, context and entity with neural networks for entity disambiguation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 1333–1339. AAAI Press.

Ledell Yu Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *EMNLP*.

Zhaohui Wu, Yang Song, and C. Lee Giles. 2016. Exploring multiple feature spaces for novel entity discovery. In *AAAI*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *NAACL*.

A Model

In the following sections, we explain our model's architecture in detail. It relies on Blink's bi-encoder architecture (680M parameters). The model can be downloaded from:

https://github.com/facebookresearch/BLINK

The code for clustering is located here:

https://github.com/rloganiv/streaming-cdc

A.1 Mention Detection

For every span [i,j], the MD head calculates the probability of [i,j] being the mention of an entity by scoring whether i is the start of the mention, j is the end of the mention, and the tokens between i and j are the insides:

$$s_{start(i)} = \mathbf{w}_{start}^T \mathbf{p}_i$$
$$s_{end(j)} = \mathbf{w}_{end}^T \mathbf{p}_j$$
$$s_{mention(t)} = \mathbf{w}_{mention}^T \mathbf{p}_t$$

where $\mathbf{w}_{start}, \mathbf{w}_{end}, \mathbf{w}_{mention}$ are learnable vectors and $\mathbf{p}_i$ paragraph token representations based on BERT:

$$[\mathbf{p}_1 \dots \mathbf{p}_n] = BERT([CLS]p_1 \dots p_n[SEP])$$

Overall mention probabilities are computed as:

$$p([i,j]) = \sigma(s_{start(i)} + s_{end(j)} + \sum_{t=i}^{j}(s_{mention(t)}))$$

Top candidates are selected as mention candidates and propagate to the next step.

A.2 Entity Disambiguation

The ED head receives mention spans in the text and finds the best matching entity in the KB.

Following Wu et al. (2020), ED is based on dense retrieval. Description-based entity representations are computed as follows:

$$\mathbf{e} = BERT_{[CLS]}([CLS]t(e)[SEP]d(e)[SEP])$$

Following Li et al. (2020), mention representations are constructed with one pass of the encoder and without mention boundary tokens by pooling mention tokens through a single feed-forward layer (*FFL*) from the encoder output:

$$\mathbf{m}_{i,j} = FFL(\mathbf{p}_i \dots \mathbf{p}_j)$$

Similarity score s between the mention candidate and an entity candidate $e \in E$ are computed:

$$s(e, [i,j]) = \mathbf{e} * \mathbf{m}_{i,j}$$

A likelihood distribution over all entities, conditioned on the mention $[i,j]$ is computed:

$$p(e|[i,j]) = \frac{exp(s(e,[i,j]))}{\sum_{e' \in E} exp(s(e',[i,j]))}$$

$< [i,j], e^* >$, such that

$$e^* = argmax_e(p([i,j],e)),$$

are passed as a candidate < mention span, entity > tuple to the rejection head.

A.3 Rejection head

MD and ED steps over-generate. R looks at an $(e^*, [i,j])$ pair holistically decides whether to accept it. Input features to R are the MD score $p([i,j])$, the ED score $p(e^*|[i,j])$, the mention representation $\mathbf{y}_{i,j}$, top-ranked candidate representation $\mathbf{x}_{e^*}$ as well as their difference and Hadamard product. The concatenation of these features is fed through a feed-forward network to output the final entity linking score $p([i,j],e^*)$. All $p([i,j],e^*) > \gamma$ are accepted where γ is a threshold set to 0.4.

A.4 Training

Following prior work (Sun et al., 2015b; Cao et al., 2018; Gillick et al., 2019; Onoe and Durrett, 2020), training is split into two stages. First, ED only is trained on a Wikipedia dataset. This dataset is constructed by extracting Wikipedia hyperlinks to labeled mention-entity pairs and consists of 17M training samples. Then, ED, MD and R are trained jointly on the downstream dataset (either Oscar or Wikipedia). Outputs from one component are fed as input to the next and losses are summed together. To train the ED head, frozen entity representations are used. As entity embeddings do not change during training, entity embeddings can be indexed using quantization algorithms for a fast kNN search (using FAISS (Johnson et al., 2017) framework with HNSW index). A likelihood distribution over positive and mined hard negative entities for each

Parameter	Value
d_m	0.8171
s_m	0.5
d_e	110

Table 5: Hyper-parameters adaptation phase

mention is computed. Negative Log-Likelihood loss across all gold mentions in the text is used.

To train MD, binary cross-entropy loss between all possible valid spans and gold mentions in the training set is computed. Valid spans are spans with begin < end, less than a maximum length, and we also filter out spans that start or end in the middle of the word.

To train R, binary cross-entropy loss between retrieved mention-entity pairs and gold mention-entity pairs is used.

Outputs from one component are fed as input to the next and losses are summed together.

B OSCAR-based dataset

OSCAR data can be downloaded here:

https://oscar-corpus.com/

We select the following six online news pages:

BBC: https://www.bbc.com/
CNN: https://www.cnn.com/
Deutsche Welle: https://www.dw.com/en/
Reuters: https://www.reuters.com/article/
Guardian: https://www.theguardian.com/
Associated Press: https://apnews.com/article/

C Hyper-parameters adaptation phase

Using OSCAR D_{adapt}-dev, we optimize mention score threshold s_m, greedy NN distance threshold d_m and mention entity similarity threshold d_e.

We optimize s_m in range 0.0 to 1.0 in steps of 0.1 for E_U discovery recall. We optimize d_m in range 0.5 to 1.0, in steps of 0.0001 for NMI. We optimize d_e for E_U discovery recall in range 50 to 250 in steps of 10. For results, see Table 5.

We report recall of 81% and precision of 6% for clusters referring to unknown entities. Recall of clusters referring to known entities is 88% with precision 96%. Clustering NMI is 0.92.

D Adapting to the updated index

We show that by re-training L after indexing, L learns to circumvent E_U: We identify known entities part of the training set that are in close proximity of unknown entities (*confusable known entities*). We compare the average similarity between mentions and their respective linked entity when adding unknown entities before training vs. after training. Mean similarity when adding unknown entities before training is 93.28 for confusable known entities and 92.57 for other known entities. A t-test shows that this difference is significant (p-value of 0.0001 with < 0.05). As a reference, mean similarity when adding unknown entities post training is 92.65 irrespective of whether they are confusable or not.

E Disambiguation Results

Besides end-to-end performance, we also report entity disambiguation performance with oracle mention detection in Table 6.

F Wikipedia Results

We report performance on Wikipedia D_{t2}-test in Table 7. Due to a smaller D_{adapt}, end-to-end performance is lower. When filtering Wikipedia D_{t2}-test for mentions of discovered entities, L_{t1}-Cluster-Oracle precision is 40.5 and L_{t1}-Cluster recall is 15.3.

G Infrastructure, Training and Inference Details

We ran all training distributed across 8 NVIDIA TESLA V100 GPUs, each with 32 GB of memory. The first training stage took 48h, the second one 12h.

Adaptation phase is currently limited by expensive greedy NN clustering with quadratic time complexity but the type of clustering is interchangeable for more efficient ones. We chose this type of clustering as Logan IV et al. (2021) showed it performs decently for BLINK based mention encodings.

	OSCAR					
	Unknown Entities			Known Entities		
Model	**R**	**P**	**NMI**	**R**	**P**	**NMI**
L_{Dt1}	0.0	0.0	0.0	92.2	92.2	96.0
L_{Dt2}	63.5	45.3	96.8	90.0	90.2	96.0
L_{t1}-Descp	58.0	33.9	96.3	92.1	92.3	96.1
L_{t1}-Mention	26.2	30.8	92.7	92.2	92.2	96.0
EDIN (L_{t1}-Cluster)	27.9	34.1	93.4	92.2	92.2	96.2

Table 6: **Entity Disambiguation performance** on OSCAR D_{t2}-test.

	Unknown Entities			Known Entities		
Model	**R**	**P**	**NMI**	**R**	**P**	**NMI**
L_{t1}	0.0	0.0	0.0	70.5	75.8	95.4
L_{t2}	33.6	25.0	98.3	70.6	75.4	95.3
L_{t1}-Descp	33.9	20.0	98.0	71.2	74.4	95.3
L_{t1}-Cluster-Oracle	7.8	55.6	90.6	70.1	75.9	95.6
EDIN (L_{t1}-Cluster)	1.8	15.4	93.4	71.1	74.1	95.3

Table 7: **End-to-end EL performance** on Wikipedia D_{t2}-test.

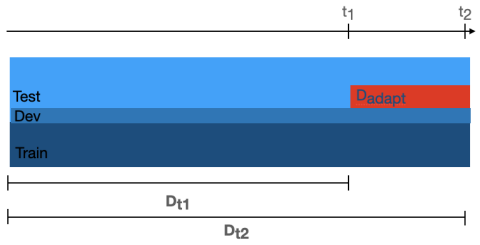

Figure 2: **Dataset splits:** A schema illustrating the composition of D_{t1} and D_{t2}. Note, that contrary to what this plot suggests, the number of samples per data split is equal for D_{t1} and D_{t2}.