DeepType 2: Superhuman Entity Linking
All You Need Is Type Interactions

Jonathan Raiman
Paris-Saclay, NVIDIA
jonathanraiman@gmail.com

Abstract
Across multiple domains from computer vision to speech recognition, machine learning models have been shown to match or outperform human experts at recognition tasks. We lack such a comparison point for Entity Linking. We construct a human benchmark on two standard datasets (TAC KBP 2010 and AIDA (YAGO)) to measure human accuracy. We find that current systems still fall short of human performance. We present DeepType 2, a novel entity linking system that closes the gap. Our proposed approach overcomes shortcomings of previous type-based entity linking systems, and does not use pre-trained language models to reach this level. Three key innovations are responsible for DeepType 2’s performance: 1) an abstracted representation of entities that favors shared learning and greater sample efficiency, 2) autoregressive entity features indicating type interactions (e.g. list type homogeneity, shared employers, geographical co-occurrence) with previous predictions that enable globally coherent document-wide predictions, 3) the entire model is trained end to end using a single entity-level maximum likelihood objective function. This is made possible by associating a context-specific score to each of the entity’s abstract representation’s sub-components (types), and summing these scores to form a candidate entity logit. In this paper, we explain how this factorization focuses the learning on the salient types of the candidate entities. Furthermore, we show how the scores can serve as a rationale for predictions.

The key contributions of this work are twofold: 1) we create the first human performance benchmark on standard benchmarks in entity linking (TAC KBP 2010 and AIDA (YAGO)) which will be made publicly available to support further analyses, 2) we obtain a new state of the art on seven standard EL datasets and attains higher than human accuracy from our benchmark on TAC and AIDA. Most of our gains are explained by type interactions: an entity representation that captures rich inter-entity relations by encoding entities using their typed Wikidata neighbors. Predictions are coherent thanks to a document-wide score trained by a contrastive loss; the score retains type-system’s explanatory power by capturing the per-type contribution to each prediction. The system also enables practical use of document coherency features by materializing them on-the-fly during search with a knowledge base in the loop.

The paper is structured as follows: Section 2 states the EL problem; Section 3 presents related work; Section 4 describes our approach; Section 5 presents experiments measuring hu-
man EL accuracy and shows how DeepType 2 profits from type-based representations, negative sampling, and global normalization; Section 6 contains a discussion of the results, a conclusion, and future work directions.

Problem Statement
The goal of entity linking is to find the exact element (entity) in the knowledge base (KB) referenced by a pre-highlighted phrase (mention) in an input document\(^1\). Using an alias table collected on training corpora we can store potential mappings between mentions and entities. The task then becomes finding the correct entity among the alias table results, rather than considering all entities in the full KB (6+ million in the English Wikipedia). In Figure 1 we see that the alias table entry for “Ada” has two possible options: Ada Lovelace or Ada language.

Related Work
The state of the art in entity identification and disambiguation can be structured along several dimensions we discuss below.

Abstract Entity Representations and Types. Ling, Singh, and Weld (2015) proposed to use the diverse types of NER tags (e.g., persons, places) to categorize all candidate entities in their EL system. The use of abstract entities was further generalized in DeepType (Raiman and Raiman 2018), considering all Wikidata classes as potential categories, or types, and shows a type predictor suffices to disambiguate. Abstract description-based representations are also used in (Nie et al. 2018; Logeswaran et al. 2019). In (Mulan’ et al. 2020), the proposed EL system combines pre-trained language models with entities described by a transcription of their Wikidata relations.

Identification and Disambiguation Loss. Most approaches rely on either generative or contrastive losses. In the former case, the sought model is optimized to maximize the log-likelihood of the ground truth interpretation. In the latter case, the model is optimized to enforce a sufficient margin between the ground truth interpretation and alternatives (Gutmann and Hyvärinen 2010). The two approaches have complementary strengths and weaknesses. The generative approach is based on first principles; it enables to assess any interpretation at the expense of a (very) high sample complexity; the challenge is to define the search space. The contrastive approach, only aims at making the good interpretation the preferred one by only requiring that the different input spaces (images, text, knowledge graph nodes) project into a mutual scalar comparison space (Nie et al. 2018).

Coherency. DeepType 2 uses a neural network that takes as input entire documents with their mentions. An illustration of the architecture is given in Figure 2.

Mention Representation For each mention we use an alias table to generate candidate entities. Our alias table is generated using the same approach as prior work (Ferragina and Scaiella 2010): intra-wiki links from Wikipedia provide a mapping from mention to linked entities. We also exploit the link statistic features from the alias table: 1) prior probability of linking to a particular entity given a particular alias table entry, 2) prior probability a given mention was seen for a given entity.

SoTA and Attention. A recent trend in Entity Linking systems is instead to perform independent predictions but use pre-trained language models with attention to ensure long-range context informs each prediction (Nie et al. 2018; Wu et al. 2019; Mulan’ et al. 2020; Ling et al. 2020). The features from language modeling help to ensure the model learns a rich textual encoding, and also reduces the chances of overfitting when transferring a model from a high supervision regime (language modeling) to a sparsely supervised setting (EL). The high memory and computation cost limit the applicability of these models to long documents. The current SoTA (Févry et al. 2020) circumvents this issue by truncating the document to keep a window around a mention. This approach approximates global context by gluing back the document title to the window.

Discussion. Several lessons are learned from the above approaches. For instance, rich classes as in DeepType (Raiman and Raiman 2018) enable a type Oracle to reach 99% accuracy on TAC and AIDA, but their type classifier does not realize this potential. Furthermore, this approach forces to coerce entities into exclusive type labels, which requires human intervention in this preprocessing step. Along these lines, our proposed approach associates entities with a variable number of type labels removing preprocessing. We switch the objective function from predicting types to entity disambiguation in a contrastive loss setting. To reduce the computational cost of coherency metrics, pairwise entity features are materialized by DeepType 2 during search by live querying a KB.

Approach

Neural Network Architecture
DeepType 2 uses a neural network that takes as input entire documents with their mentions. An illustration of the architecture is given in Figure 2.
We obtain for each document-mention pair \( D_m \) a fixed-length representation \( h_m(D_m) \) from the variable number of mention tokens. First we max-pool the associated Bi-LSTM hidden states (3 in Figure 2) to produce \( h_{pool,m}(D_m) \). Second, we obtain longer-range context using QKV Attention (Vaswani et al. 2017) (\( \text{Att}(\cdot) \)) over the full document with \( h_{pool,m}(D_m) \) as query, and linearly project the pooled and attended vector into the same space via a learnt matrix \( W \) to obtain \( h_m \):

\[
h_m(D_m) = W \cdot h_{pool,m}(D_m) + \text{Att}(h_{pool,m}(D_m)).
\]

**Entity Representation** Next, we associate to each candidate entity multiple sets of Wikidata neighbors (e.g. human, United Kingdom, mathematician) coming from different typed relations such as occupation or origin country. These neighborhood relations are chosen based on usage frequency in Wikidata. See Appendix for the list of type neighborhood relations. The neighbors obtained from these relations can be entities, real values (e.g. latitude/longitude), or dates (e.g. birthdate). We refer to neighbors that are up to \( n_{neighborhood\_depth} \) steps away as the type neighborhood representation of an entity. See Figure 1 for an example of Ada Lovelace’s type neighborhood.

To recover a fixed length entity representation from the type neighborhood we use a Graph Neural Network (GNN). In this work \( n_{neighborhood\_depth} = 2 \), which enables us to take advantage of a basic GNN consisting of an embedding layer and a max-pool. For deeper neighborhoods, a depth or edge-aware GNN might be preferable (Wang et al. 2019).

**Type Interactions** We perform joint predictions over all mentions in a document. In order to do this, we augment the entity representation with two sets of features related to past predictions: latent and discrete type interactions.

Latent type interactions are obtained by computing the scalar product between the type neighborhood representation of a candidate and the hidden state of a decoder LSTM (6. in Figure 2). The decoder LSTM receives as input the
chosen entity’s type neighborhood representation after each prediction. Latent interactions measure if the candidate’s type neighborhoods match the memory using a learnt function.

Discrete type interactions are boolean features corresponding to the result of multiple knowledge graph queries. For each relation in a predefined set, a knowledge graph query checks if any past entity is connected to the candidate entity by this relation. Using these features it is possible to measure list type-homogeneity or answer questions such as “is this candidate of the same sport / team / league/ etc. as past entities?” As with type neighborhoods, relations were chosen based on their Wikidata usage frequency. As we later discuss in Section 4, certain relations are redundant, and the system is robust to removing those. See Appendix for the list of Wikidata relations used in the type interactions. The discrete interactions access outside information from a KB to answer factual inter-entity questions. We provide in Figure 3 an example of these interactions to disambiguate John Gorst.

**Score** Candidate probabilities are obtained from the dot product between the mention and the entity representation: with \(c_0, \ldots, c_n\) candidate entities, \(s\) the discrete/latent state, and \(f_k(c_i, D_m, s)\) the concatenation of type neighborhood and interaction features:

\[
Score(c_i, D_m, s) = h_m(D_m) \cdot f_k(c_i, D_m, s),
\]

\[
P(c_i|D_m, s) = \frac{\exp \left( \text{Score}(c_i, D_m, s) \right)}{\sum_{j=0}^{n} \exp \left( \text{Score}(c_j, D_m, s) \right)}.
\]

**Rewriting** \(\text{Score}(c_i, D_m, s)\) as a sum of feature scores (see Appendix) reveals each type neighborhood or interaction’s contribution to the overall score. Feature scores may serve as decision justifications as we show in Figure 1.

**Objective Function** Model parameters \(\theta\) are learnt by minimizing \(\mathcal{L}(\theta)\), the negative log likelihood of the ground truth entity \(e\) relative to alias table candidates for the mention \(m\):

\[
\mathcal{L}(\theta) = \sum_{(e,D_m,s)} - \log P(e|D_m, s; \theta).
\]

**Contrastive Loss** Our objective function profits from becoming a contrastive loss. When too many candidates are returned by the alias table we subsample to reduce computational cost, and when there are too few, we supply negative samples (Gutmann and Hyvärinen 2010). Negative samples massively increase the supervision signal as over 45.4% of Wikipedia mentions are unambiguous.

A further reason to use a contrastive loss is its ability to focus model capacity towards only resolving actual ambiguities from the alias table. By comparison, a generative loss for predicting types independently wastes capacity on modeling all type combinations (e.g. \(2^{128}\) in DeepType (Raiman and Raiman 2018)) most of which are impossible. A contrastive loss focuses the learning on discriminative features; the gradient is zero for features common between candidates (proof in Appendix).

**Densification**

In order to observe type interaction features we densify mentions in documents. For training, we densify Wikipedia articles by creating new links to entities already present in the page. We filter new links with a classifier trained on 300 hand-collected labels. As articles do not refer to themselves, the subject of the article can be used to create many additional links. Keeping the high confidence new links increases dramatically the size of our training corpus by 2.96x from 74M to 220M mentions. Details for our classifier are given in the Appendix.

**Coherency**

The New Yorker's Steve Coll looked into more than $54 million allegedly stashed...

Figure 4: Global normalization effect in TAC: Steve Coll, although unambiguous, reinforces the likelihood of picking his employer New Yorker magazine when scores are summed before being normalized.

To make coherent predictions we jointly predict entities while taking into account interactions between all predicted entities:

- Discrete type interactions act as constraints to prune the candidate search space: in the context of syntactic structures such as “Venice, California” we expect a located-in relation, or “Paris and London” where we expect list type homogeneity.
- Beam search with autoregressive features increases coherency with limited computational cost: DeepType 2’s pairwise entity features are only materialized
during search. For \( k \) search beams, and document \( D \) with \( N_D \) input tokens, \( N_m \) mentions, \( M_c \) candidates per mention, and \( N_m \cdot N_D \) attention features, the computational complexity is \( O(N_m \cdot M_c \cdot k + N_m \cdot N_D) \) instead of \( O(N_m^2 \cdot M_c^2) \) if all features had to be pre-computed. The practical implications is that this system can process all AIDA with 16 search beams in 23s (187.3 mentions/s and 2178 tokens/s) on an NVIDIA GeForce GTX 1080.

- Global normalization enables every decision, regardless of order, to increase or decrease the joint likelihood of the prediction sequence. This is achieved by summing all decision scores before normalizing, rather than multiplying locally normalized probabilities as shown in Figure 4. This conversion from local to global was previously proposed to improve expressivity and overcome label bias (Andor et al. 2016; Raiman and Miller 2017), an autoregressive model pitfall (Lafferty, McCallum, and Pereira 2001).

### Results

We first seek to identify the remaining gap between humans and algorithms on EL by establishing a human performance benchmark. Second, we evaluate DeepType 2 on standard benchmarks. Third, we investigate through ablations what aspects of the proposed approach are the most important.

In all our experiments DeepType 2 is trained for 2 million gradient steps using as annotations intra-wiki links from the December 2017 English Wikipedia dump with densification, as well as AIDA’s train split. Unless otherwise noted, we use 16 search beams and global normalization. Training takes approximately 6 days on a single NVIDIA GTX 1080Ti on a computer with 128GB of RAM and 28 core 3.3Ghz Intel i9 CPUs. To facilitate comparisons with prior work on AIDA we use the PPR4NED alias table (Pershina, He, and Grishman 2015), otherwise our alias table is built from intrawiki links. Neural network hyperparameters were selected using a Wikipedia-based validation set and are given in the Appendix. Our code is available at this url\(^2\). The Wikipedia dump is downloadable at this url\(^3\) and is licensed under CC BY-SA and GFDL. AIDA’s train split is freely available for research purposes from NIST\(^4\).

#### Human Performance Benchmark

We measure human performance using a panel of annotators from Amazon Mechanical Turk (AMT) on TAC and AIDA. We find that humans reach 96.86% on TAC and 96.78% on AIDA, outperforming the current state of the art on these tasks as shown in Table 2.

To improve response quality we take particular care to screen, brief, and provide proper incentives to the annotators: 1) they must have AMT’s Master qualification, a recognition of prior excellence in annotation tasks, 2) we give a bonus for correct answers to align incentives, 3) annotators are screened by testing that they read the instructions and requiring they

\(^2\)https://github.com/deep-type/deeptype2
\(^3\)https://dumps.wikimedia.org
\(^4\)https://trec.nist.gov/data/reuters/reuters.html

reach a minimum accuracy on a trial subset of the data. To further reduce the effect of annotator expertise differences, each labelled mention is assigned to 3 different annotators and we measure accuracy using the best provided response (Oracle). Inter-annotator agreement (Fleiss’ \( \kappa \)) is high, supporting the belief that annotators reached similar conclusions and did not respond randomly. A summary of our results are given in Table 1, details of the AMT annotation interface are given in the Appendix, and we release the annotations from this benchmark at this url\(^5\).

#### Evaluation on Standard Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>DeepType 2 (ours)</th>
<th>Yang et al. (2018)</th>
<th>De Cao et al. (2020)</th>
</tr>
</thead>
<tbody>
<tr>
<td>W-CWEB</td>
<td>85.57±0.24</td>
<td>81.8</td>
<td>77.3</td>
</tr>
<tr>
<td>W-WIKI</td>
<td>87.83±0.08</td>
<td>79.2</td>
<td>87.4</td>
</tr>
<tr>
<td>MSNBC</td>
<td>95.12±0.23</td>
<td>92.6</td>
<td>94.3</td>
</tr>
<tr>
<td>AQUIAINT</td>
<td>92.74±0.27</td>
<td>89.9</td>
<td>89.9</td>
</tr>
<tr>
<td>ACE 2004</td>
<td>92.23±0.19</td>
<td>89.2</td>
<td>90.1</td>
</tr>
</tbody>
</table>

Table 3: EL system accuracy on standard datasets (\( \mu \pm \sigma \)).

We compare Human performance, DeepType 2, and the current EL state of the art on the standard benchmark datasets TAC and AIDA and report our results with average and standard deviation across 6 runs in Table 2. In Table 3 we report evaluations of our system on five additional well known EL datasets WNED-WIKI (Guo and Barbosa 2018), WNED-CWEB (Guo and Barbosa 2018), MSNBC (Cucerzan 2007), AQUIAINT (Milne and Witten 2008), and ACE 2004 (Ratinov et al. 2011).

We first find that humans outperform existing approaches, suggesting that there is room for algorithms to improve. Humans have similar accuracy on TAC and AIDA, while sur-
prisingly SoTA algorithmic approaches until 2020 perform 4.09% higher on AIDA than TAC. DeepType 2 improves accuracy over the SoTA on all evaluated datasets, and outperforms the human oracle accuracy by 0.62% on TAC and 0.74% on AIDA. The largest gains relative to prior work are observed on TAC (2.58%), AIDA (1.02%), WNED-CWEB (3.77%), while the smallest is WNED-WIKI (0.43%).

**Mention Densification** One of the largest gains relative to prior work is observed on TAC, greatly thanks to the way mention “densification” provides additional contextual entities that power type interaction: we add mentions to the document to increase their frequency from TAC’s original single mention/document. Mentions are detected by greedily taking the longest alias table matches linkable to persons, places, or activities. Accuracy increases by 3.97% from 93.51% to 97.48%.

<table>
<thead>
<tr>
<th>Decision Method</th>
<th>TAC</th>
<th>AIDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Independent</td>
<td>93.51±0.07</td>
<td>96.76±0.08</td>
</tr>
<tr>
<td>Joint Local Score</td>
<td>97.44±0.08</td>
<td>97.62±0.07</td>
</tr>
<tr>
<td>Joint Global Score</td>
<td>97.48±0.06</td>
<td>97.72±0.04</td>
</tr>
</tbody>
</table>

Table 4: Impact (μ ± σ) of decision method on accuracy.

<table>
<thead>
<tr>
<th>k</th>
<th>TAC</th>
<th>AIDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>97.44±0.08</td>
<td>97.69±0.06</td>
</tr>
<tr>
<td>8</td>
<td>97.44±0.08</td>
<td>97.71±0.04</td>
</tr>
<tr>
<td>16</td>
<td>97.48±0.06</td>
<td>97.72±0.04</td>
</tr>
</tbody>
</table>

Table 5: Impact (μ ± σ) of varying search beams k on accuracy.

**Joint Decision Making** The score given to a sequence of predictions is heavily dependent on type interaction features to make coherent decisions. We report the result of independent predictions versus joint predictions in Table 4. We observe a massive improvement over independent decisions when jointly predicting entities. A smaller but noticeable improvement is visible when switching from locally to globally normalized scores.

We also study the effect of varying the number of search beams in Table 5. We find that a small percentage of search errors in TAC and AIDA can be mitigated by considering more hypotheses.

**Error Analysis**

DeepType 2 has the ground truth entity in its top-3 responses over 99% of the time (99.10% on TAC, 99.35% on AIDA). The main remaining mistakes made by DeepType 2 and humans fall into the same category: confusing places and sports teams due to journalistic shorthand overloading the meaning of place names as visible in Table 6.

**Ablations**

**Entity Representation** The comparison of different entity representations in DeepType 2 shows that the best one uses both type neighborhoods and type interactions as visible in Table 7. We empirically verify the effect of replacing type neighborhoods by same dimension unique Entity-Vectors (EV) used in SoTA approaches (Yamada et al. 2016; Sil et al. 2018; Ling et al. 2020; Févry et al. 2020). Type neighborhoods have 6 times less parameters (166M vs. 998M), get the same accuracy as EV after training on a 1/4 of data and 10 times less updates (150k vs. 1.5M), and reach higher accuracy model on TAC and AIDA. We also ablate the use of type interactions and find that they also contribute to a large portion of the EV system’s performance.

**Type Interaction Features** Our entity representation ablation above shows type interactions are crucial, begging the question: what are the most important type interactions? We compare the impact of using a single type interaction on TAC and AIDA accuracy in the (A) pyramid plot in Figure 5. We observe that type interactions are domain-dependent: relations such as “League” matter more in sports-heavy AIDA, and geographical relations (e.g. “Located in”) benefit the newswire-based TAC.

As type interactions can have overlapping roles, we look at the sensitivity to removing a single type interaction as an indication of its redundancy and report the results in the (B) pyramid plot of Figure 5. “Located in” has the largest negative impact when removed and thus is least redundant. Conversely, “League” appears redundant as it individually increases AIDA accuracy by 0.58% accuracy, but only causes a 0.12% decrease if removed when all other type interactions are present.

**Training Ablations**

**Wikipedia Densification** We compare the quality of models trained with and without densification. With densification models obtain higher accuracy on TAC and AIDA as shown.
Figure 5: Type interactions are domain dependent as visible in (A) by looking at the impact of using a single relation in TAC vs. AIDA. In (B) we test the redundancy of type interactions by removing one from the system.

Table 8: Impact of Wikipedia Densification on accuracy.

<table>
<thead>
<tr>
<th>Training data</th>
<th>TAC</th>
<th>AIDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>96.75</td>
<td>96.47</td>
</tr>
<tr>
<td>Densified</td>
<td>97.48</td>
<td>97.72</td>
</tr>
</tbody>
</table>

Table 9: Negative sampling impact on EL performance.

<table>
<thead>
<tr>
<th>Negative Samples</th>
<th>Max candidate entities/mention if training</th>
<th>TAC</th>
<th>AIDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>20</td>
<td>95.41</td>
<td>95.63</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>96.98</td>
<td>97.99</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>97.48</td>
<td>97.72</td>
</tr>
</tbody>
</table>

The performance gains are explained by a novel abstract entity representation built on Wikidata relation subgraphs. Through ablations we show that this entity representation uses 80% fewer parameters than equivalent entity vectors, and reaches higher accuracies thanks to an ability to share learning between entities of the same type. The strongest contributor to performance is the set of autoregressive relational features we call type interactions. These features enable the system to produce coherent document-wide predictions through higher order reasoning over the entity types (e.g. shared employers, geographical co-occurrence, list type homogeneity). A further benefit of DeepType 2 is that it eliminates two major difficulties of existing type-based systems such as DeepType (Raiman and Raiman 2018): 1) the type representation is now automatically generated by embedding subgraphs rather than curated type labels, 2) a single task-aligned objective function replaces prior use of a proxy multi-objective type classification.

Our work has several limitations. First, there is a need to extend the human benchmark by broadening it to additional datasets and languages. Second, DeepType 2 relies solely on structured relations and cannot make use of the wealth of unstructured relations. Third, the presented system DeepType 2 does not take advantage of pre-trained language models. A useful line of investigation would be to test the effect of pre-training and alternate text encoding mechanisms.

**Discussion and Conclusion**

We establish an Entity Linking human benchmark that measures human performance and provides a milestone for algorithmic approaches. The principal difficulty in setting up this benchmark is to obtain high quality responses from annotators. We use trial runs and bonuses for correct answers to obtain high inter-annotator agreement and accurate responses. We further reduce the effect of annotator expertise differences by measuring the best human response (oracle). Through this benchmark, we observe that previous systems approach human performance but still underperform.

We close the performance gap thanks to a new EL system, DeepType 2. The proposed approach removes the need for a pre-trained language model and improves over the human accuracy on the benchmark datasets and reaches a new state of the art on five other commonly used EL datasets.

**References**


