Headword-Oriented Entity Linking: A Special Entity Linking Task with Dataset and Baseline

Mu Yang[†], Chi-Yen Chen[†], Yi-Hui Lee[†], Qian-Hui Zeng[†], Wei-Yun Ma[*†], Chen-Yang Shih[‡], Wei-Jhih Chen[‡]

[†]Institute of Information Science, Academia Sinica, Taipei, Taiwan
[‡]R&D Center, PIXNET Corporation, Taipei, Taiwan
emfomy@gmail.com, chiyench@usc.edu, yi-hui.lee@utdallas.edu, littlemiss330@gmail.com, ma@iis.sinica.edu.tw, kent@pixnet.tw, wayne@pixnet.tw

Abstract

In this paper, we design headword-oriented entity linking (HEL), a specialized entity linking problem in which only the headwords of the entities are to be linked to knowledge bases; mention scopes of the entities do not need to be identified in the problem setting. This special task is motivated by the fact that in many articles referring to specific products, the complete full product names are rarely written; instead, they are often abbreviated to shorter, irregular versions or even just to their headwords, which are usually their product types, such as "stick" or "mask" in a cosmetic context. To fully design the special task, we construct a labeled cosmetic corpus as a public benchmark for this problem, and propose a product embedding model to address the task, where each product corresponds to a dense representation to encode the different information on products and their context jointly. Besides, to increase training data, we propose a special transfer learning framework in which distant supervision with heuristic patterns is first utilized, followed by supervised learning using a small amount of manually labeled data. The experimental results show that our model provides a strong benchmark performance on the special task.

Keywords: Corpus, Information Extraction, Distant Supervision

1. Introduction

Named entity linking, or entity linking (EL), is determining the identity of entities that are mentioned in the text and bridging them from unstructured textual data with structural knowledge bases. Traditionally, entity linking involves two sequential tasks: first, detecting mentions using named entity recognition, seeking to locate entity chunks in text and classify their entity types; second, linking the recognized entity mention with the corresponding entry in a knowledge base.

Domain-specific entity linking has attracted attention due to commercial demands in practice, especially product name linking. Unlike conventional named entities such as person, location, or organization names, complete full product names are rarely written in context; instead, they are often abbreviated to a shorter, irregular version or even just to their headwords. For instance, in a real blog article about "Dior Addict Lacquer Stick" (PID 17755 in our cosmetic database) and other cosmetic products, the 'PID 17755' item is sometimes written as "Dior lacquer stick", or just "pen-shaped lacquer stick" or even abbreviated to its product type — "stick", such as "I really like the stick that Dior released last month," where "stick" refers to 'PID 17755'. In most cases, the mentions contain at least the product type to represent the product. Such being the case, from a practical application perspective such as product recommendation, the key goal is to link each product-type word, such as "stick", "mask", "eyeshadow", etc., to knowledge bases if they do represent certain products in the context without the need for named entity recognition.

Based on this concept, in this paper, we design headword-oriented entity linking (HEL), a specialized entity linking problem in which only the headwords of the entities are to be linked to knowledge bases; the mention scopes of the entities do not need to be identified in the problem setting. HEL can be seen as a distinctive integration of traditional EL and coreference resolution.

Figure 1: Headword-Oriented Entity linking — only link "唇釉" (Stick, the red text in the figure) to an unique knowledge base ID, instead of linking its whole named entity — "DIOR癮誘超模漆光唇釉" (Dior Addict Lacquer Stick, the green background in the figure) — to the ID.

To fully design this special task, we construct a labeled cosmetic corpus as a public benchmark[1] for this problem. We collect blog articles from PIXNET[2] and seek to link cosmetic products in the articles to the cosmetics database, as shown in Fig. 1. Most cosmetics are not written in their

[*]Corresponding author

[1]The package and dataset are available at https://github.com/ckiplab/cosmel.

[2]One of the largest online Taiwanese blog sites and social networking service companies.

full product names. Word deletions, replacements, and insertions are very common in these articles. Also, since not every product-type word represents a specific product, we use two other labels — one is **GP** (general product), representing an uncertain product, such as "I never use sticks". The other is **OSP** (other specific products), meaning it is indeed a particular product but is not listed in our database (no entry in the database).

In this paper, we also propose a product embedding model for this task. Each product corresponds to a dense representation that jointly encodes not only its context but also various information, including its full product name and official advertised description.

Additionally, to increase training data, a special transfer learning framework is applied — several heuristic patterns are first used to generate noisy labeled data for all articles; these are then used to distantly supervise the model (Mintz et al., 2009) as a pretraining phase. After that, fully supervised learning based on a small amount of manually labeled data is further applied to fine-tune the model.

Our contributions are five-fold:

- We design headword-oriented entity linking, a special named linking problem in which mention scopes are not required to be identified. This setting is especially useful for product linking.
- We create a labeled cosmetic corpus as a public benchmark for this problem.
- We propose a product embedding model to address the problem as a strong baseline, where diverse information about products is jointly encoded using dense representations.
- We present a special transfer learning framework, involving distant supervision of the model with a large number of noisy labels, and then supervised learning using a small number of manually labeled data.
- To the best of our knowledge, we are the first to study entity analysis in the cosmetic domain.

2. Related Work

The two main strategies for entity linking are discrete feature-based and embedding-based systems. Shen et al. (2015) systematically organize discrete feature-based entity linking systems. Discrete features include name string comparisons (Liu et al., 2013) and similarities of bags of words (Lin et al., 2012), concept vectors (Chen and Ji, 2011), and other manually designed features (McNamee et al., 2009). With these discrete features, supervised learning models such as binary classifiers (Zhang et al., 2010) and learning-to-rank methods (Kulkarni et al., 2009) have been implemented for entity ranking.

However, featured-based entity linking systems are highly data-dependent, which requires extensive effort to design domain-specific features. Moreover, as the discrete features are often too sparse to train the model, they are unlikely to apply to different domains. Recently, many approaches have devised to learn representations of the entity and use it for entity linking. Gupta et al. (2017) build an embedding-based linking system that learns representations for each entity without domain-specific training data or hand-engineered features. Yamada et al. (2016) learn word and entity embeddings for named entity disambiguation based on the skip-gram model. Francis-Landau et al. (2016) utilize convolutional neural networks to capture semantic correspondence between a mention's context and a proposed target entity. Sun et al. (2015) disambiguate entities using the mention embedding (the average of the embeddings of the words it contains), the context embedding by a convolutional neural network (CNN), and entity embedding through entity surface words and entity class from the knowledge base.

Conventionally, supervised models have been used for entity linking. However, one significant problem with supervised approaches is their heavy reliance on large amounts of annotated training data. Moreover, entity linking annotation is expensive and time-consuming. Some supervised approaches train their models on a small manually-created data set consisting of thousands of labeled entity mentions (Shen et al., 2012; Dredze et al., 2010; McNamee, 2010; Li et al., 2009). Some systems (Bunescu and Paşca, 2006; Agirre et al., 2009) use hyperlinks in Wikipedia articles to construct training data.

For all these entity linking researches, mention scopes need to be identified by named entity recognition, but the scope identification is not always indispensable in terms of practical needs for applications, such as product recommendation or product analysis. That motivates us to design this special task based on headword-oriented entity linking in this paper.

Our special task can be regarded as nominal coreference (Fonseca et al., 2018) and traditional entity linking. Such mentions with complex lexicon syntactic patterns are linked together as coreference chains before being linked to unique data or knowledge base IDs. One can also use that pipeline to tackle this problem, but in our design, we do not distract developer from using pipeline — we proposed a knowledge base driven approach to tackle entity-linking without explicit coreference resolution directly.

3. Corpus Construction

We collected over $5{,}000$ products and about $50{,}000$ blog articles from the PIXstyleMe[3] web service. These articles contain a total of over 5 million sentences and over 41 million words. We standardized the text in the database and applied word segmentation. We also created repositories of products, brands, and product types as knowledge bases.

3.1. Product Database

Product We created a product database from PIXstyleMe's database consisting of product names, brands, descriptions, etc. After manually removing duplicate products and fixing typos, we collected $5{,}060$ products in the database, and assigned a unique ID (product ID, PID) for each product.

[3]A fashion community of PIXNET, `https://styleme.pixnet.net`

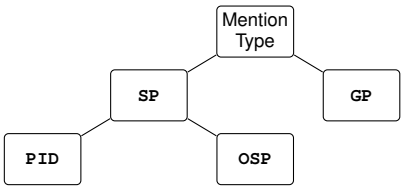

Figure 2: Mention types: **SP** (Specific Product), **PID** (Product with ID), **OSP** (Other Specific Product), and **GP** (General Product)

Brand We also collected the aliases (including English and Chinese names) of each brand (364 in total) from the database. For example, both "sk2" and "skii" refer to the brand "SK-II"; "shuuemura", "shu uemura", and "植村秀" represent the brand "Shu Uemura".

Headword We collected 926 headwords, each of which represents a type of product. Examples are "面膜" (facial mask), "唇膏" (lipstick), and "香水" (perfume).

Word Segmentation After the above preprocessing, we applied word segmentation using CKIPWS (Ma and Chen, 2003a; Ma and Chen, 2003b) on product names and their descriptions. In order to get better word segmentation, we add many cosmetic specific words into our lexicon.
We also make some necessary modifications to the ambiguous terms after word segmentation. For example, "修容蜜粉餅" can be segmented as "修容" (contouring) + "蜜粉餅" (pressed powder cake) or "修容蜜" (liquid blush) + "粉餅" (powder cake). Though the previous one is more semantically reasonable, the word segmentation tool is not able to determine it since all the above four words are collected in the lexicon. Therefore, we add "修容蜜粉餅" to the lexicon, and segment it using regular expressions after word segmentation. We added about a hundred such terms into our dataset. With the above trick, all the headwords in product names are segmented correctly. This is a crucial step for mention detection (§3.2.).

3.2. Mention Detection

In conventional entity linking, most entities have a specific scope and can be differentiated as people, time, or organizations. However, cosmetic entities are different from general named entities. To be more specific, most cosmetic entities are long noun phrases, such as "迪奧輕透光空氣蜜粉" (PID 6064, Diorskin, Nude Air Loose Powder). Also, most of the time, while being mentioned, entities do not appear as the full name — only 20,617 of a total 967,969 mentions are written as complete full product names. For instance, both "輕透光空氣蜜粉" (nude air loose powder) and "空氣蜜粉" (air loose powder) can be linked to the product "迪奧輕透光空氣蜜粉" (Diorskin Nude Air Loose Powder).

Mention Type As shown in Fig. 2, we classify the mentions into several types: **PID**, **OSP**, and **GP**. **PID** (Product with ID) mentions are linked to specific products in the database. **OSP** (Other Specific Product) mentions are also specific products but are not contained in the database. **GP** (General Product) mentions are general concepts or plural forms. **OSP** and **GP** correspond to **NIL** mentions in conventional entity linking.

Dataset	Total	PID	OSP	GP
RLabel	906,585	94,826	195,307	616,452
GLabel	40,970	5,778	11,469	23,723

Table 1: Dataset mentions

3.3. Headword-Oriented Entity Linking

To reflect the different characteristics of cosmetic data, we redefine the scope of entities. Instead of detecting whole noun phrases, we simply detect the headword of the phrase, for instance, "蜜粉" (loose powder) for the above example. In this way, 967,969 mentions are detected in our corpus.
We called this method headword-oriented entity linking (HEL). With this special idea, the time spent in the annotation of the corpus will be much less than conventional entity linking. This argument would stand for any product types besides cosmetics domain. That means it will be much easier and faster to prepare a training dataset for a new domain. Without HEL, annotators need to read all sentences (over 5 million sentences) in the corpus, whereas our annotators only need to read those sentences containing headwords (only 800K sentences). Please note that our goal is to let the special task only focus on relatively easy but the most popular, critical cases, in order to prepare the training set for a new domain quickly. Aliases without headword are left to other approaches/tasks, such as coreference resolution, to identify.

3.4. Annotation and Heuristic Rule

Although there is no well-labeled data set in the cosmetics domain, it is not practical to manually label the entire corpus. To increase training data, a special transfer learning framework is presented. In essence, we first design several simple heuristic patterns (Heuristic Rule, a rule-based method) using regex and string matching based on observation to generate noisy labeled data for all articles, denoted as the *RLabel* corpus, and then use them to supervise the model in a pretraining phase distantly.
After that, to fine-tune the model, we further apply fully supervised learning based on a small amount of manually labeled data, denoted as the *GLabel* (golden label) corpus.
As an example of a heuristic pattern, noun phrases after "這款" (this; note that "款" is a quantifier in Chinese that widely used for cosmetic products) are likely to be cosmetic products. For example, the phrase "這款面膜" (this facial mask) usually refers to a facial mask product occurred previously, while "一款面膜" (a facial mask) is generally used as a general concept.
The distributions of the *RLabel* and *GLabel* corpora are given in Tab. 1.

4. Product Embedding Model

We seek to build a baseline model for the special task. The model aims to link each cosmetic product's headword to the corresponding product entry in the knowledge base. If it refers to a product not in our database, it is labeled **OSP**; if it is a general concept, it is marked **GP**.
To jointly consider various features of products, we follow the idea of joint encoding (Gupta et al., 2017) and present

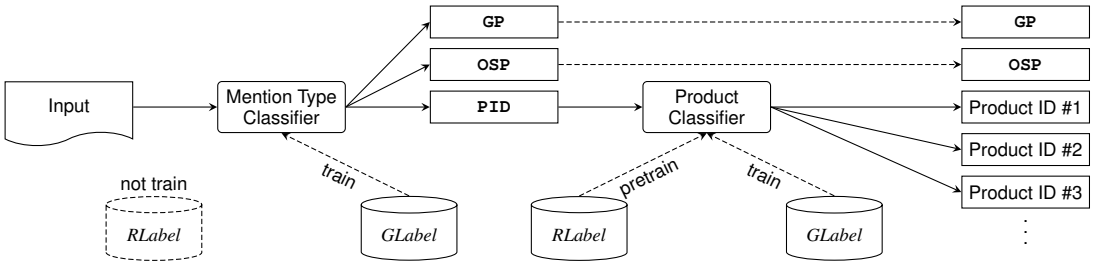

Figure 3: The pipeline — the mention type classifier (§4.1.) and the product classifier (§4.2.).

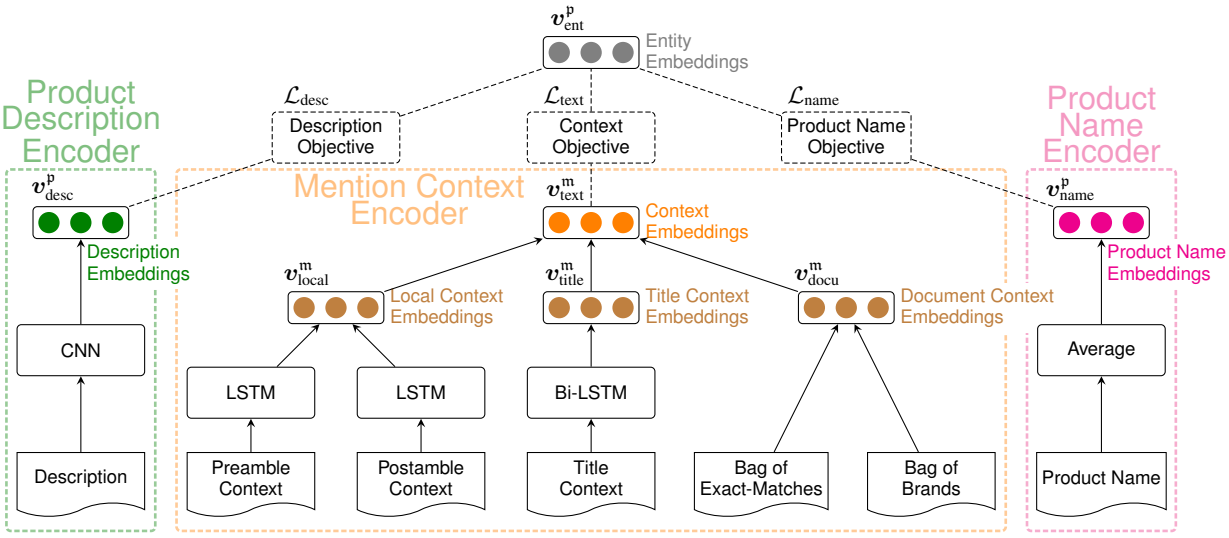

Figure 4: Product classifier (§4.2.) and its modules — the product description encoder (**D**, §4.3.2.), the mention context encoder (**C**, §4.3.1.), and the product name encoder (**N**, §4.3.3.).

a product embedding model for our problem. Each product is represented as a dense representation to jointly encode products' context and various information about the products, including their advertised description and product names. We define $\mathcal{P}$ as the set of cosmetic products, $\mathcal{B}$ as the set of brands, $\mathcal{T} = \{\texttt{PID}, \texttt{OSP}, \texttt{GP}\}$ as the set of mention types, $\mathcal{M}$ as the set of mentions, and $\mathcal{V}$ as the set of vocabularies.

In our experiment, we found that a single model is not able to solve the problem, as the number of mentions labeled **OSP** and **GP** covers around 90% of the mentions, whereas **PID** contains 5,060 different products but only covers around 10% of the mentions; thus the model would be forced to pay more attention to the recall of **GP**/**OSP** even sacrificing the precision of other labels (the products with ID). With a single model only, it would gain a total accuracy of 77%, with 91% precision on **GP**, 78% on **OSP**, and 0% on all product IDs. However, product IDs should be more important for practical needs.

To address this problem, a pipeline strategy (Fig. 3) is introduced: we first use a mention type classifier (§4.1.) to classify the mentions into the three mention types, and then apply a product classifier (§4.2.) to link further those **PID** mentions to cosmetic products in our database.

4.1. Mention Type Classifier

With the mention type classifier, we seek to distinguish **PID** (products with predefined ID), **OSP** (products not in our database), and **GP** (general concept).

Here we use a context encoder (§4.3.1.) to obtain the mention context embeddings $\boldsymbol{v}^{\mathfrak{m}}_{\text{text}}$ of a given mention $\mathfrak{m} \in \mathcal{M}$, and linearly project the embeddings onto a 3-dimensional vector $\boldsymbol{v}^{\mathfrak{m}}_{\text{mtype}} = (u^{\mathfrak{m}}_{\texttt{PID}}, u^{\mathfrak{m}}_{\texttt{OSP}}, u^{\mathfrak{m}}_{\texttt{GP}})^\top \in \mathbb{R}^3$, and apply cross-entropy loss.

4.2. Product Classifier

As shown in Fig. 4, to jointly encode the information of products, we employ three encoders: a mention context encoder (§4.3.1.), a product description encoder (§4.3.2.), and a product name encoder (§4.3.3.). For more information, see §4.3.

Given a product $\mathfrak{p} \in \mathcal{P}$, and any $\mathfrak{m}$ linked to this product, our target is to acquire a representation of the entity $\boldsymbol{v}^{\mathfrak{p}}_{\text{ent}} \in \mathbb{R}^D$ that is similar to the mention context embeddings $\boldsymbol{v}^{\mathfrak{m}}_{\text{text}}$, the product description embeddings $\boldsymbol{v}^{\mathfrak{p}}_{\text{desc}}$, and the product name embeddings $\boldsymbol{v}^{\mathfrak{p}}_{\text{name}}$ obtained by the above three encoders.

Precisely, we maximize the probability of predicting the correct product $\mathfrak{p}_{\mathfrak{m}} \in \mathcal{P}$ from a given mention $\mathfrak{m}$ as

$$P_{\text{text}}(\mathfrak{p}_{\mathfrak{m}}|\mathfrak{m}) = \frac{\exp\left(\boldsymbol{v}^{\mathfrak{p}_{\mathfrak{m}}}_{\text{ent}} \cdot \boldsymbol{v}^{\mathfrak{m}}_{\text{text}}\right)}{\sum_{p \in \mathcal{P}} \exp\left(\boldsymbol{v}^{p}_{\text{ent}} \cdot \boldsymbol{v}^{\mathfrak{m}}_{\text{text}}\right)}, \tag{1}$$

and maximize the log-likelihood

$$\mathcal{L}_{\text{text}} = \frac{1}{|\mathcal{M}|} \sum_{\mathfrak{m} \in \mathcal{M}} \log P_{\text{text}}(\mathfrak{p}_{\mathfrak{m}}|\mathfrak{m}). \tag{2}$$

We use an objective similar to that above for product description and name embeddings; that is, we maximize the probabilities $P_{\text{desc}}(\mathfrak{p}|\boldsymbol{v}^{\mathfrak{p}}_{\text{desc}})$ and $P_{\text{name}}(\mathfrak{p}|\boldsymbol{v}^{\mathfrak{p}}_{\text{name}})$, defined similarly as eq. (1), and maximize the log-likelihoods $\mathcal{L}_{\text{desc}}$ and $\mathcal{L}_{\text{name}}$, defined similarly as eq. (2).
Finally, we maximize the summation of the objectives

$$\mathcal{L}_{\text{ent}} = \mathcal{L}_{\text{text}} + \mathcal{L}_{\text{desc}} + \mathcal{L}_{\text{name}} \tag{3}$$

to learn the product embeddings $\boldsymbol{v}^{\mathfrak{p}}_{\text{ent}}$ for each product $\mathfrak{p} \in \mathcal{P}$.

4.3. Encoder Modules

In this section, we describe several encoder modules used in our model. Following the idea of joint encoding in Gupta et al. (2017), we modify the mention context encoder (§4.3.1.) and the product description encoder (§4.3.2.). Furthermore, we propose an additional product name encoder (§4.3.3.) inspired by Sun et al. (2015).

4.3.1. Encoding the Mention Context — C

For mention context, we encode local, title, and document contexts, and combine them into mention embeddings.

Local-Context Encoder The sentence containing a mention is usually the most important for the corresponding cosmetic product. It might contain the name of the product and some description of it.
Given a mention word $\mathfrak{m} \in \mathcal{M}$, which denotes the sentence contains this mention as $s = (w_1, \ldots, \mathfrak{m}, \ldots, w_L)$, where $w_\bullet$ are the words and L is the length of this sentence, we split the sentence into the preamble $\overrightarrow{s} = (w_1, w_2, \ldots, \mathfrak{m})$ and the postamble[4] $\overleftarrow{s} = (w_L, w_{L-1}, \ldots, \mathfrak{m})$. We apply two different long short term memory networks (LSTM) (Hochreiter and Schmidhuber, 1997) to both preamble and postamble contexts separately, using the pretrained word embeddings of each word as inputs. We concatenate the last hidden states $\overrightarrow{\boldsymbol{h}}_{\mathfrak{m}}, \overleftarrow{\boldsymbol{h}}_{\mathfrak{m}} \in \mathbb{R}^d$ of both LSTMs and pass them through a single-layer feed-forward network to produce the local context embeddings $\boldsymbol{v}^{\mathfrak{m}}_{\text{local}} \in \mathbb{R}^D$.

Title-Context Encoder The title of an article usually contains the article's topic. Most mentions in this article are related to this topic.
Similar to the local-context encoder, we use LSTM on the title of the article containing the given mention $\mathfrak{m}$. Here we use bidirectional LSTM, concatenate the last hidden states of both directions, and pass them through a single-layer feed-forward network to produce title context embeddings $\boldsymbol{v}^{\mathfrak{m}}_{\text{title}} \in \mathbb{R}^D$.

Document-Context Encoder We use document-wise information to obtain document context embeddings. The first is the *exact match*. An exact match is a scope of characters in the article which uses exactly the same characters as one of the cosmetic product. We assume that the mention $\mathfrak{m}$ is related to one or more products mentioned in the article. Since we have high confidence in the exact matches, such information is valuable to producing embeddings. We use a bag-of-products representation $\boldsymbol{v}^{\mathfrak{m}}_{\text{exact}} \in \{0,1\}^{|\mathcal{P}|}$, similar to Lazic et al. (2015), when collecting the exact-matched products that appeared in the article.
Also, we assume that the mention $\mathfrak{m}$ is related to one of the brands mentioned in the article. Similarly, we use a bag-of-brands representation $\boldsymbol{v}^{\mathfrak{m}}_{\text{brand}} \in \{0,1\}^{|\mathcal{B}|}$, collecting all the brands that appear in the article.
To produce the document context embeddings $\boldsymbol{v}^{\mathfrak{m}}_{\text{docu}} \in \mathbb{R}^D$, we concatenate $\boldsymbol{v}^{\mathfrak{m}}_{\text{exact}}$ and $\boldsymbol{v}^{\mathfrak{m}}_{\text{brand}}$ and pass them through a single-layer feed-forward network.

Mention-Context Encoder Finally, we combine the above local $\boldsymbol{v}^{\mathfrak{m}}_{\text{local}}$, title $\boldsymbol{v}^{\mathfrak{m}}_{\text{title}}$, and document $\boldsymbol{v}^{\mathfrak{m}}_{\text{docu}}$ context embeddings by concatenating and passing them into a single-layer feed-forward network. The output embeddings are denoted as $\boldsymbol{v}_{\text{text}} \in \mathbb{R}^D$, containing all the information on the given mention $\mathfrak{m}$ (Fig. 4, middle).

4.3.2. Encoding the Product Description — D

The textual description of a cosmetic product provides its ingredients, usage, effect, and features. This information helps us to produce embeddings with cosmetic product knowledge bases.

Product Description Encoder Given a product $\mathfrak{p} \in \mathcal{P}$, denote the description of this product as $d = (w_1, \ldots, w_L)$. Similar to Francis-Landau et al. (2016), we apply a convolution neural network (CNN) on the sentence d, followed by a maximum pooling layer, using the pretrained word embeddings of each word as inputs. The CNN outputs are passed into a single-layer feed-forward network to produce the product description embeddings $\boldsymbol{v}^{\mathfrak{p}}_{\text{desc}}$ (Fig. 4, left).

4.3.3. Encoding the Product Name — N

The name of a cosmetic product is one of the unique features in cosmetic product entity linking. Unlike most entity linking, the cosmetic product names are usually very long, and are thus useful for entity recognition.

Product Name Encoder Denote the name of a product $\mathfrak{p}$ as $n = (w_1, \ldots, w_L)$. We average the pretrained word embeddings of each word $w_\bullet$ to obtain the product name embeddings $\boldsymbol{v}^{\mathfrak{p}}_{\text{name}}$ (Fig. 4, right).

4.4. Transfer Learning Based on Distant Supervision

To increase the amount of training data, we present a special transfer learning framework: we first distantly supervise the model with a large number of noisy labels (*RLabel*), and then use supervised learning with a small number of manually labeled data (*GLabel*). *RLabel* contains more data but is less reliable, whereas *GLabel* is more reliable but the size is limited.
In the experiments, the complete process is denoted as *RLabel+GLabel*; *RLabel* refers to using only noisy labels, and *GLabel* refers to using only manually labeled data.
We expect the *RLabel+GLabel* model to be able first to learn a big picture framed by the simple patterns and then fine-tune the model to capture recognition details provided by the golden labels. This assumption is validated in the following experiments.

[4] The postamble context is reversed so that the LSTM starts at the last word w_L and ends at the mention $\mathfrak{m}$.

Metric		HEL			Baseline
		Dataset			Rule
		RLabel	*GLabel*	*RLabel+GLabel*	
Overall	*Accuracy*	66.31±0.23	**84.15**±0.46	80.79±0.54	65.17
	F1 Score	64.29±0.18	**84.16**±0.38	80.88±0.47	63.89
PID	*F1 Score*	61.34±0.35	**76.39**±0.83	76.29±0.58	62.39
OSP	*F1 Score*	37.54±0.30	**77.29**±0.50	72.01±0.47	38.49
GP	*F1 Score*	77.35±0.20	**89.14**±0.41	86.04±0.55	75.98

Table 2: Accuracy and F-measure (mean and standard deviation, %) of mention type classifier

Metric		Modules	HEL			Baseline			
			Dataset			Similarity Classifier			Rule
			RLabel	*GLabel*	*RLabel+GLabel*	*Emb*	*Bag*	*Emb+Bag*	
PID	*Accuracy*	**C**	65.94±0.66	78.96±0.65	85.14±0.57	58.00	56.07	57.06	53.20
		C+D	65.55±0.34	79.71±0.73	84.62±0.36				
		C+N	65.66±0.69	79.81±0.94	**87.51**±0.52				
		C+D+N	65.38±0.64	79.41±0.92	86.43±0.65				
	F1 Score	**C**	64.83±0.56	76.94±0.71	83.43±0.64	62.78	61.20	62.09	57.06
		C+D	64.34±0.39	77.71±0.71	82.82±0.47				
		C+N	64.99±0.77	77.74±0.96	**86.52**±0.66				
		C+D+N	64.55±0.64	77.30±1.05	85.47±0.65				

Table 3: **PID** accuracy and F-measure (mean and standard deviation, %) of product classifier. Here we test only data labeled with a PID.

Metric		Modules	HEL				Baseline
			Dataset for Mention Type Classifier				
			GLabel		*RLabel+GLabel*		Rule
			Dataset for Entity Embeddings Model				
			GLabel	*RLabel+GLabel*	*GLabel*	*RLabel+GLabel*	
Overall	*Accuracy*	**C**	82.77±0.55	83.21±0.51	79.34±0.55	79.96±0.53	64.92
		C+D	82.82±0.55	83.20±0.53	79.42±0.55	79.93±0.53	
		C+N	82.81±0.59	**83.40**±0.51	79.43±0.53	80.18±0.53	
		C+D+N	82.81±0.54	83.34±0.52	79.39±0.52	80.12±0.49	
	F1 Score	**C**	82.48±0.35	82.92±0.31	79.16±0.46	79.79±0.45	62.44
		C+D	82.53±0.32	82.91±0.32	79.24±0.49	79.76±0.45	
		C+N	82.51±0.36	83.20±0.31	79.23±0.47	80.11±0.49	
		C+D+N	82.51±0.34	**83.34**±0.30	79.19±0.43	80.04±0.43	

Table 4: Overall accuracy (mean and standard deviation, %) of joint model of mention type classifier and product classifier

5. Experiments

Dataset We use two datasets: *RLabel* for rule-labeled IDs (noisy labels) with 906,585 mentions, and *GLabel* for human-labeled IDs (golden labels) with 40,970 mentions. We split both datasets into training and test subsets at a ratio of 7 : 3, and extract 30% data from the training set for validation. The database contains $|\mathcal{P}| = 5{,}060$ products with $|\mathcal{B}| = 364$ brands, and the vocabulary size was $|\mathcal{V}| = 86{,}873$.

Word Embeddings Pretraining We apply the skip-gram model (Mikolov et al., 2013) to the corpus to obtain word embeddings. Since we collect the brand aliases as mentioned in §3.1., we average the embeddings of the brand aliases for each brand.

Hyper-parameters We use $D = 300$ dimensional pretrained word embeddings, and $d = 100$ dimensional vectors for LSTM and CNN hidden layers. The output embeddings of each encoder module and the product embeddings were also $D = 300$ dimensions. The CNN window size was 5. We use the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.001, mini-batches of size 32, and 10 epochs.

Evaluation Setup For each model, we train it on *RLabel*, *GLabel*, and *RLabel+GLabel* (§4.4., pretrain on *RLabel* and then train on *GLabel*), respectively, and then evaluated it on the *GLabel* testing dataset. We evaluate each model 10 times and compute the means and standard deviations of the accuracies and F1 scores. Here we average the F1 scores weighted by their supports (the number of true

instances for each label).

5.1. Baselines

Due to the lack of existing models that perfectly fit out special task HEL, we use two simple baselines — the heuristic rule (§3.4.) and the similarity classifier (described below). The similarity classifier computes the similarity of mentions with their candidate products. Note that the similarity classifier can be used only for product ID linking; that is, it can be used only as a baseline of product classification.

Similarity Classifier Given a mention $\mathtt{m} \in \mathcal{M}$, we select a candidate product $\mathtt{p} \in \mathcal{P}$ with the same headword as the mention. We find a noun phrase n by sentence parsing (Hsieh et al., 2007; Hsieh et al., 2012; Yang et al., 2008) the mention with the most significant Jaccard similarity coefficient[5] to the candidate, and compute the similarity of $\mathtt{p}$ and n. Precisely, we propose two similarity methods — embeddings (denoted as *Emb*) and bag-of-words (denoted as *Bag*). First, we use the product name encoder (§4.3.3.) on both the product name $\mathtt{p}$ and the noun phrase n, and compute the cosine similarity $s^{\mathtt{p},n}_{\text{emb}}$ of those embeddings. We assign the ID of the candidate with the highest similarity to the mention. Another way is to replace the above embeddings by bags of words to compute $s^{\mathtt{p},n}_{\text{bag}}$. Besides, we also evaluate the accuracy by adding the above similarities $s^{\mathtt{p},n}_{\text{emb}} + s^{\mathtt{p},n}_{\text{bag}}$ (denoted as *Emb+Bag*).

5.2. Evaluation of Mention Type Classifier

We evaluate the mention type classifier using the *RLabel*, *GLabel*, and *RLabel+GLabel* datasets, and compare the accuracies and F1 scores with the baselines (the heuristic rule, §3.4.). We also compute the F1 scores of the three mention types **`PID`**, **`OSP`**, and **`GP`**. From Tab. 2, we find that the model using *GLabel* outperforms those using other datasets. We conclude that the performance of the models affected by the heuristic rule (*RLabel* and *RLabel+GLabel*) does not meet our expectations, as the heuristic rule performs poorly on **`OSP`**.

5.3. Evaluation of Product Classifier

We test the product classifier with different encoder modules. The model using the context encoder (§4.3.1.) is denoted as **C**, that using the product description encoder (§4.3.2.) as **D**, and that using the product name encoder (§4.3.3.) as **N**. Since the context is necessary, we test all combinations with context — **C**, **C+D**, **C+N**, and **C+D+N**. We evaluate the above models using the *RLabel*, *GLabel*, and *RLabel+GLabel* datasets, and compare the accuracies and F1 scores with the baselines (the heuristic rule, §3.4., and the similarity classifiers, §5.1.). Here we use only the mentions labeled in **`PID`** for both training and testing.

Interestingly, the model using the *RLabel* dataset outperforms *RLabel* itself (the heuristic rule) by evaluating both accuracy and F-measure on *GLabel* even if we exclude the product databases (that is, model **C**, the model using the context encoder only). Since the heuristic rule uses a decision tree for labeling, we believe it is too arbitrary. However, the model produces embeddings for each product, uses similarity to determine the label, and avoids overdetermination.

[5]The Jaccard similarity coefficient of two sets A and B, also called the Jaccard Index, is defined as $|A \cap B|/|A \cup B|$.

5.4. Joint Evaluation of Mention Type Classifier and Product Classifier

Finally, we join the mention type classifier and product classifier for an end-to-end evaluation. We first apply the mention type classifier on all mentions in the testing data, and apply the product classifier on these labeled **`PID`** by mention type classifier to yield a specific ID. We further investigate the models using *GLabel* and *RLabel+GLabel* datasets only, as the performance of *RLabel* was relatively week. In Tab. 4, we compare the four combinations of both models with both datasets (*GLabel* and *RLabel+GLabel*).

We find that the model using *GLabel* on the mention type classifier and using *RLabel+GLabel* on the entity embeddings model achieves the best results. This is intuitive, as the mention type classifier and the entity embeddings model perform best with *GLabel* and *RLabel+GLabel*, respectively. Although the **C+N** model outperforms other encoder module combinations in terms of accuracy, the joint model using all encoder modules (**C+D+N**) yields the best F1 score.

6. Conclusion

The paper defines headword-oriented entity linking, a specialized entity linking problem in which only the headwords of the entities are to be linked to knowledge bases without named entity recognition. We create a labeled cosmetic corpus as a public benchmark for this problem, and propose a product-embedding model as a strong baseline to solve it, which simultaneously takes into account various types of context information and the information related to the products themselves. Moreover, we present a special transfer learning framework based on distant supervision. The model and the data set are also released to the public as the benchmark.

We believe HEL will fill many practical commercial needs, such as product recommendation and data mining for products.

7. Acknowledgments

We are grateful for the insightful comments from anonymous reviewers. This work is supported by PIXNET Inc. and the Ministry of Science and Technology of Taiwan under grant numbers 109-2634-F-001-010.

Bibliographical References

Agirre, E., Chang, A. X., Jurafsky, D., Manning, C. D., Spitkovsky, V. I., and Yeh, E. (2009). Stanford-UBC at TAC-KBP. In *TAC*.

Bunescu, R. and Paşca, M. (2006). Using encyclopedic knowledge for named entity disambiguation. In *11th conference of the European Chapter of the Association for Computational Linguistics*.

Chen, Z. and Ji, H. (2011). Collaborative ranking: A case study on entity linking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 771–781. Association for Computational Linguistics.

Dredze, M., McNamee, P., Rao, D., Gerber, A., and Finin, T. (2010). Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 277–285. Association for Computational Linguistics.

Fonseca, E., Vanin, A., and Vieira, R. (2018). Nominal coreference resolution using semantic knowledge. In *International Conference on Computational Processing of the Portuguese Language*, pages 37–45. Springer.

Francis-Landau, M., Durrett, G., and Klein, D. (2016). Capturing semantic similarity for entity linking with convolutional neural networks. *arXiv preprint arXiv:1604.00734*.

Gupta, N., Singh, S., and Roth, D. (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.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hsieh, Y.-M., Yang, D.-C., and Chen, K.-J. (2007). Improve parsing performance by self-learning. *International Journal of Computational Linguistics & Chinese Language Processing*, 12(2):195–216.

Hsieh, Y.-M., Bai, M.-H., Chang, J. S., and Chen, K.-J. (2012). Improving PCFG chinese parsing with context-dependent probability re-estimation. In *Proceedings of the Second CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 216–221.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kulkarni, S., Singh, A., Ramakrishnan, G., and Chakrabarti, S. (2009). Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM.

Lazic, N., Subramanya, A., Ringgaard, M., and Pereira, F. (2015). Plato: A selective context model for entity resolution. *Transactions of the Association for Computational Linguistics*, 3:503–515.

Li, F., Zheng, Z., Bu, F., Tang, Y., Zhu, X., and Huang, M. (2009). THU QUANTA at TAC 2009 KBP and RTE track. In *TAC*.

Lin, T., Etzioni, O., et al. (2012). Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 84–88. Association for Computational Linguistics.

Liu, X., Li, Y., Wu, H., Zhou, M., Wei, F., and Lu, Y. (2013). Entity linking for tweets. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1304–1311.

Ma, W.-Y. and Chen, K.-J. (2003a). A bottom-up merging algorithm for chinese unknown word extraction. In *Proceedings of the second SIGHAN workshop on Chinese language processing*, volume 17, pages 31–38. Association for Computational Linguistics.

Ma, W.-Y. and Chen, K.-J. (2003b). Introduction to CKIP chinese word segmentation system for the first international chinese word segmentation bakeoff. In *Proceedings of the second SIGHAN workshop on Chinese language processing*, volume 17, pages 168–171. Association for Computational Linguistics.

McNamee, P., Dredze, M., Gerber, A., Garera, N., Finin, T., Mayfield, J., Piatko, C. D., Rao, D., Yarowsky, D., and Dreyer, M. (2009). HLTCOE approaches to knowledge base population at tac 2009. In *TAC*.

McNamee, P. (2010). HLTCOE efforts in entity linking at tac kbp 2010. In *TAC*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mintz, M., Bills, S., Snow, R., and Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Shen, W., Wang, J., Luo, P., and Wang, M. (2012). Linden: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st international conference on World Wide Web*, pages 449–458. ACM.

Shen, W., Wang, J., and Han, J. (2015). Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.

Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., and Wang, X. (2015). Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*, pages 1333–1339.

Yamada, I., Shindo, H., Takeda, H., and Takefuji, Y. (2016). Joint learning of the embedding of words and entities for named entity disambiguation. *arXiv preprint arXiv:1601.01343*.

Yang, D.-C., Hsieh, Y.-M., and Chen, K.-J. (2008). Resolving ambiguities of chinese conjunctive structures by divide-and-conquer approaches. In *Proceedings of the Third International Joint Conference on Natural Language Processing*.

Zhang, W., Su, J., Tan, C. L., and Wang, W. T. (2010). Entity linking leveraging: automatically generated annotation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1290–1298. Association for Computational Linguistics.