UOM-FJ at the NTCIR-15 SHINRA2020-ML Task

Hiyori Yoshikawa
Fujitsu Laboratories Ltd., Japan
y.hiyori@fujitsu.com

Chunpeng Ma
Fujitsu Laboratories Ltd., Japan
ma.chunpeng@fujitsu.com

Aili Shen
University of Melbourne, Australia
ailis@student.unimelb.edu.au

Qian Sun
University of Melbourne, Australia
qiasun@student.unimelb.edu.au

Chenbang Huang
University of Melbourne, Australia
chenbangh@student.unimelb.edu.au

Guillaume Pelat
École Polytechnique, France
guillaume.pelat@polytechnique.edu

Akiba Miura
Fujitsu Laboratories Ltd., Japan
miura.akiba@fujitsu.com

Daniel Beck
University of Melbourne, Australia
d.beck@unimelb.edu.au

Timothy Baldwin
University of Melbourne, Australia
tbaldwin@unimelb.edu.au

Tomoya Iwakura
Fujitsu Laboratories Ltd., Japan
iwakura.tomoya@fujitsu.com

ABSTRACT

The NTCIR-15 SHINRA2020-ML Task is a multilingual text categorization task where a Wikipedia page in a given language is mapped to one or more of the 219 Extended Named Entity (ENE) categories. The UOM-FJ team participated in 28 out of the 30 subtasks (languages), with a primary focus on English. Our system makes use of different types of information associated with the target Wikipedia articles, as well as the hierarchical structure of ENE. Our system ranked first on the English subtask with an F1-score of 82.73, demonstrating the effectiveness of using different types of document information.

TEAM NAME

UOM-FJ

SUBTASKS

English, Spanish, French, German, Chinese, Russian, Portuguese, Italian, Arabic, Indonesian, Turkish, Dutch, Polish, Persian, Vietnamese, Korean, Hebrew, Romanian, Norwegian, Czech, Ukrainian, Hindi, Finnish, Hungarian, Danish, Thai, Catalan, Bulgarian

1 INTRODUCTION

The NTCIR-15 SHINRA2020-ML Task [13] is a multilingual document categorization task aimed at constructing a structured knowledge base based on Wikipedia. The task is to classify entities represented by Wikipedia articles in different languages into the 219 Extended Named Entity (ENE) categories [1]. Participants are provided with target Wikipedia articles, as well as labelled training data for each target language. The labels in the training data come from corresponding Japanese articles, i.e., Japanese articles are categorized first and then the labels are propagated to the corresponding articles in other languages using inter-language links. While the resulting dataset is potentially noisy, it is large enough to employ supervised training over each of the associated monolingual datasets to create a classifier for each target language. Further details of the task can be found in the task overview paper [13].

We participated in 28 out of the 30 subtasks (languages),[1] with a primary focus on English. Our system makes use of different types of information associated with Wikipedia to obtain document representations. A Wikipedia article contains not only a textual description of the target entity but also other data modalities including images relevant to the entity, page links connecting relevant entities to each other, and an infobox that summarizes key attributes of the entity. By incorporating such information, we expect to obtain richer document representations to help categorize entities into the fine-grained ENE types. Furthermore, we utilize the hierarchical structure of the ENE taxonomy, to capture similarities between classes and potentially boost data efficiency. In order to aggregate different types of information efficiently, we separately compute document embeddings for individual modalities of input. The embeddings are computed by different sub-models based on different aspects of the input document and then combined into a multi-aspect document representation. Our system ranked first in four languages including English, demonstrating the effectiveness of our approach. This paper describes the details of our method and discusses the results.

2 OUR APPROACH

Our primary focus is on the English subtask. Despite the fact that our approach is essentially language-agnostic, we did not have enough time and resources to apply our full model to other languages. Instead of applying the fully integrated model, we adopted a simplified approach based on VL-BERT [17] for non-English subtasks. Below we first describe our main model for English, and then describe the model for the non-English subtasks in Section 2.9.

2.1 System Overview

An overview of our main model (for English) is shown in Figure 1. We cast the problem as a multi-label classification problem. That

[1]The languages we didn't participate in were Greek and Swedish. For Greek, the Wikipedia dump was not provided in JSON format, which made it hard for us to process the data. For Swedish, we could not make a submission in time because of a bug in our output generation procedure.

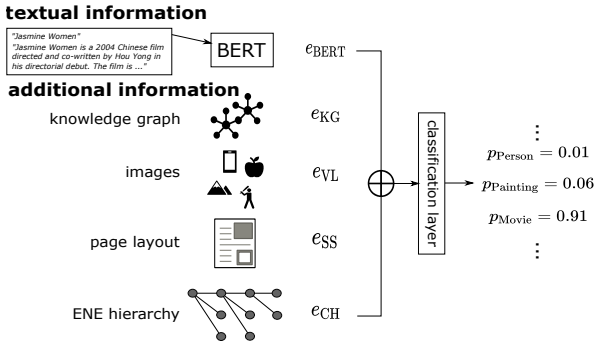

Figure 1: Our system overview.

is, for an input Wikipedia article, the system may predict multiple ENE categories, in line with the fact that the training data includes multi-label instances. Given an input article and document encoder for a given data modality, our system first encodes it into a single vector. We then feed it into a classification layer and compute a score for each candidate ENE category.

The input encoding module consists of two parts: a trainable BERT [4] encoder, and a set of fixed pre-trained document representations. To fully exploit not only the textual description but also rich multimodal information in Wikipedia, we train sub-models that use different types of information associated with Wikipedia articles to generate a document representation that reflects multiple aspects of Wikipedia articles. We describe the details of the sub-models in Sections 2.2 to 2.5. In the classification layer, the scores are computed independently, i.e., we do not take the hierarchical structure of the ENE categories into account and model each category independently. However, as described in Section 2.5, we incorporate the information of class hierarchy in computing the input document embeddings. The final model architecture is detailed in Section 2.6.

2.2 Knowledge Graph Embeddings

We use pre-trained embeddings of the Wikidata graph published by PyTorch-BigGraph [11] team [2] as our knowledge graph embeddings e_{KG}. The embeddings are trained on the full Wikidata graph[3] using PyTorch-BigGraph with a translation operator. The embeddings file contains 200-dimensional embeddings of 78 million entities. We use the `wikibase_item` field of the Wikipedia cirrus dump to identify the Wikidata entity corresponding to a Wikipedia article. We found Wikidata entities for 98.3% of all target English Wikipedia articles. We map articles with no corresponding Wikidata entity onto a shared randomly generated vector $e_{\text{KG}}^{\text{UNK}}$.

2.3 Text–image Embeddings

Generally a Wikipedia page includes one or more images, which we utilize in our approach. We propose a cross-modal approach to calculate text–image embeddings e_{VL}.

[2]https://github.com/facebookresearch/PyTorch-BigGraph
[3]The exact version of Wikidata used for training is not provided.

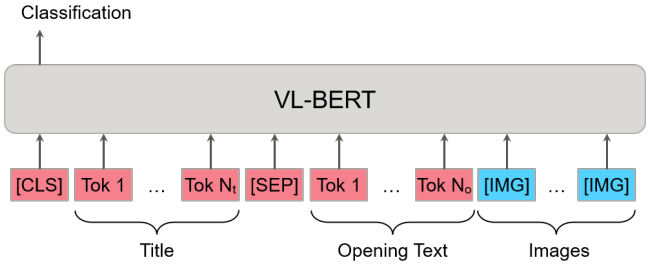

Figure 2: Model architecture of VL-BERT for SHINRA2020-ML task.

2.3.1 Model Architecture. Following the popularity of BERT and the paradigm of pre-training & fine-tuning, we chose VL-BERT [17] as our backbone model.

Inputs to VL-BERT consist of four parts: token embeddings, visual feature embeddings, segment embeddings, and position embeddings. Their summations are fed into a transformer [20]. We refer the readers to the original paper for more details.

We adapt the above model to our specific task. The model architecture is shown in Figure 2. For the text modality, the input consists of the title and opening text.[4] Textual embeddings are from pre-trained BERT model, which is also fine-tuned during training.

Specifically, we use `bert-large-uncased`[5] for English. For the image modality, instead of using regions-of-interest (ROIs) as inputs, we feed the whole image to the model, using Faster R-CNN [12] + ResNet-101 [7] to extract image features. When there are no images for a given Wikipedia page, we artificially generate an empty black image. When there are multiple images, we rescale them to the same size and concatenate them as one large image. Different components are discriminated using ROIs. Parameters are initialized from parameters pre-trained on Visual Genome [10] for object detection [2], and fine-tuned during training.

2.3.2 Data Preparation. To build the image corpus, we extract images from the English Wikipedia dump of June 2020[6] using the `zim` library.[7] There are more than 14 million pages in this dump, from which more than 5.8 million images are extracted. We inject these images into the SHINRAML-2020 dataset for each language. For the English corpus, about 88% of pages are augmented with images.

2.3.3 Model Training. For the English task, we initialize the parameters of VL-BERT with the official pretrained model[8] that is trained on two cross-modal tasks: "masked language modeling with visual clues" and "masked ROIs classification with linguistic clues". Pre-training corpora include Conceptual Captions [14], BookCorpus [22], and English Wikipedia data.

We fine-tune the model on SHINRA2020-ML. The classifier is a one-layer feed-forward neural network with 1,024 hidden nodes.

[4]The maximum number of words in the opening text is 300 tokens, and we truncate in the case that the text is longer than that.
[5]https://huggingface.co/transformers/pretrained_models.html
[6]https://dumps.wikimedia.org/other/kiwix/zim/wikipedia/wikipedia_en_all_maxi_2020-06.zim
[7]https://github.com/openzim/libzim
[8]https://github.com/jackroos/VL-BERT/tree/master/model/pretrained_model

We predict one category for each Wikipedia page, and minimize the cross-entropy loss function.

2.4 Page Screenshot Layout Embeddings

In this section, we first describe how we obtain visual renderings of Wikipedia articles, followed by a brief description of the Inception V3 model (Inception: Szegedy et al. [18]), a widely-used pre-trained visual model. Then we present how we train the Inception model for our task.

2.4.1 Screenshot Generation. To avoid extensive requests to online Wikipedia, which is discouraged by Wikipedia, we use an offline Wikipedia dump, released in June 2020, to obtain visual renderings of Wikipedia articles.[9] A visual representation of each article is generated via a 1,000×4,000-pixel screenshot of the Wikipedia article via a PhantomJS script over the rendered offline version of the article,[10] which is later resized to 500×500-pixel to feed into a visual model.

2.4.2 Inception V3. There are a wide range of models for image classification, such as VGG [16], ResNet [7], Inception V3 [18], Xception [3], and EfficientNet-Lite4 [19]. In this work, we use Inception V3 pre-trained on ImageNet[11] to obtain visual representations of the Wikipedia screenshots, based on the method of Shen et al. [15] in the context of document quality assessment.

As it is difficult to decide what types of convolution filters to apply to each layer (such as 3×3 or 5×5), the basic Inception model applies multiple convolution filters in parallel and concatenates the resulting features, which are fed into the next layer. Inception is a hybrid of multiple Inception models with different convolution filters, with the benefit of capturing both local features via smaller convolutions and abstracted features via larger convolutions.

2.4.3 Model Training. We feed resized 500×500 Wikipedia article screenshots into Inception, achieving 2,048 dimensional document embeddings e_{SS}.

2.5 Class Hierarchy-aware Embeddings

Considering the tree-based taxonomic hierarchy of ENE categories, we employ a hierarchy-aware global model (HiGAM) [21] to capture the label hierarchy and label correlations. HiGAM firstly uses a text encoder to extract textual embeddings, and then enhances those embeddings with pair-wise label information using a structure encoder.

2.5.1 Model Architecture. We once again employ `bert-large-uncased` as the text encoder, and take the output of the `[CLS]` token for each document as the textual embedding. Then we construct a hierarchy convolutional graph network (Hierarchy-GCN) [9], in which each node represents an ENE category (including both internal nodes and leaf nodes), and each directed edge represents either hierarchical relational information or correlation information. Hierarchy-GCN is initialized by a weighted adjacency matrix based on prior probabilities of parent–child relations and pairwise co-occurrences among nodes in the taxonomic hierarchy, which are computed according to the provided labelled dataset.

Following the hierarchical text feature propagation approach [21], Hierarchy-GCN takes textual embeddings as input and encodes them into node-wise embeddings based on the associated neighborhood of each node, which are then concatenated as hierarchy-aware embeddings e_{CH}.

2.5.2 Label Space. In the official dataset, other than 33 pages, all labelled Wikipedia pages are classified as leaf ENE categories in the taxonomic hierarchy. To enable hierarchical classification, we expand the original flat label space to a hierarchical label space, and label each page by all nodes on the path from the root to the corresponding leaf node, e.g. [1.4.2] → [1, 1.4, 1.4.2].

We compute binary cross-entropy loss over the hierarchical label space with recursive regularization [6] to parameters of the final fully-connected layer during training. However, we use the trained hierarchy-aware embeddings e_{CH} for classification over the flat label space in the ensemble model to ensure each page is classified to leaf nodes.

2.6 Final Classification Model

As mentioned above, the final classification model takes two types of input document representation: a document representation e_{BERT} generated by a trainable BERT encoder, and fixed pre-trained document representations e_{KG}, e_{VL}, e_{SS} and e_{CH} described above. We used the title and opening text as an input to the BERT encoder. Following a `[CLS]` token, we simply concatenate the title and opening text and add a `[SEP]` token to the end of the input. As a result, we obtain input tokens as (`[CLS]`, `tokens(title)`, `tokens(opening text)`, `[SEP]`). We set the maximum length of the input tokens to 256. We use the output of the first `[CLS]` token as the BERT document representation e_{BERT}. The BERT representation is concatenated with the pre-trained document representations and fed into a two-layer feed-forward network with ReLU activation to obtain an integrated document representation $e \in \mathbb{R}^{200}$:

$$e = \text{FFNN}(e_{\text{BERT}} \oplus e_{\text{KG}} \oplus e_{\text{VL}} \oplus e_{\text{SS}} \oplus e_{\text{CH}}). \quad (1)$$

The document representation is used to compute the probability scores of the candidate ENE categories:

$$p(c, e) = \sigma(w_c^\top e + b_c), \quad (2)$$

where σ denotes the sigmoid function, and w_c and b_c are category-specific weight and bias parameters for category c.

2.7 Training

The official training set of the English subtask contains 439,351 articles annotated with ENE categories. To train our model, we randomly sampled 10,000 articles as our internal development set, and used the remaining 429,351 articles as our training set. In addition, we found that the original training set has a duplicate page ID issue whereby more than one occurrence of the same page ID is included because different Japanese Wikipedia articles may have inter-language links to the same article in the target language. Following the instructions of the organizers, we removed duplicated page entries from the training set, and merged their gold ENE labels.

[9] We assume that entities assigned to Wikipedia articles remain unchanged, although there are minor differences between Wikipedia articles of different versions.
[10] https://github.com/ariya/phantomjs/blob/master/examples/rasterize.js
[11] http://www.image-net.org/

As a result, we obtained a training and development split consisting of 428,927 and 9,980 articles, respectively.

For final model training, we optimized the parameters of the BERT encoder and the classification layer, while keeping the pretrainined embeddings e_{KG}, e_{VL}, e_{SS} and e_{CH} fixed. We used `bert-base-cased` model to initialize our trainable BERT model. Dropout rates of 0.1 and 0.3 were applied to the BERT layers and the feed-forward network, respectively. We used Adam [8] with an initial learning rate of 5e-5 and a gradient clipping threshold of 5.0. The minibatch size was set to 8 during training. Training was continued until no improvement was observed for 10 epochs in terms of F1 score on the development set, with a maximum of 200 epochs. Our model is implemented using AllenNLP [5].

2.8 Inference

At inference time, we use the probability scores computed by Equation (2), and output the categories whose estimated probability exceed the threshold $\theta = 0.5$ as the model prediction. This submission is labelled "`jointrep`" in the results table.

For the other two submissions, we further apply a post-processing step. A problem in the original output is that it may produce outputs with no predicted label, in cases where the probability scores of all candidate categories are below the 0.5 threshold. In this case, the official evaluation regards the article to be assigned the label `IGNORED`. According to the taxonomy of the ENE [1], however, `CONCEPT` should be used to represent anything other than entities which have one of the other specific ENE categories. Therefore, we apply a post-processing step in which we add a `CONCEPT` class when no category is predicted by our system. This submission is labelled "`jointrepPostprocess`".

Our third submission is based on an ensemble of different predictions made by the five models, trained on different subsets of our training split. We further divide our training split into five parts and train five sub-models, each of which uses four parts as training data and the remaining one as development data. We then aggregate the predicted outputs by taking the union of the results from these five sub-models. After that, we apply the same post-processing step described above; this submission is labelled "`jointrepUnionPostprocess`".

2.9 Models for Non-English Subtasks

As introduced in Section 2.3, VL-BERT is an important part of the proposed model. For English tasks, as mentioned above, we integrate e_{VL} with other embeddings calculated by the different submodels, and use the integrated embedding for classification.

For non-English subtasks, instead of integrating different embeddings, we resort to VL-BERT only, trained in the same way as for English with two major differences. First, we use the `bert-base-multilingual-uncased` model, with a vocabulary size of 105, 879, and number of nodes in hidden layers changed to 768. Second, because there are no pretrained VL-BERT models available and there are no non-English datasets for VL-BERT pretraining, we fine-tune our model on the SHINRA2020-ML task directly, without pretraining.

Note that for image datasets of non-English languages, we still extract images from the English dump, because we assume that generally there are more images in English Wikipedia pages than those of other languages. We firstly use the link information between Wikipedia pages of different languages to find the correspondence between English Wikipedia pages and non-English Wikipedia pages, and then inject images from English pages to corresponding non-English language pages. For the French subtask, we additionally developed a BERT-based model using text and Wikipedia categories of the input article, and text from its incoming links as input text. We took the union of the output of that model and VL-BERT, and filtered out predictions with scores under 0.5.

3 RESULTS AND DISCUSSION

3.1 Main Results

For the final submission, participants were asked to predict the labels for all articles in a given language (*target data*). The official evaluation was based on a subset of the target data (*test data*), and participants were not notified which articles in the target data were used as the test data. In addition, a subset of the target data was provided as the *leaderboard data*, as a guide for system development on the official leaderboard page.[12] The performance is evaluated using micro-averaged F1 score.

Table 1 shows the results on the English subtask. Our three submissions ranked first, second, and third place in terms of F1-score, demonstrating the effectiveness of our approach. The best run among the three submissions was `jointrep`, which achieved an F1-score of 82.73%. While the systems applying the post-processing step described in Section 2.8 (`jointrepPostprocess` and `jointrepUnionPostprocess`) performed better over the leaderboard data, they achieved slightly lower precision on the test data, indicating that post-processing is not effective in general. The ensemble strategy used in `jointrepUnionPostprocess` slightly improved the recall, but at the expense of precision.

Table 2 shows the results on all 28 languages we participated in. In addition to English, our system also ranked first on Spanish, Italian, and Catalan, despite the substantially simplified method using only images and text. On the other hand, our system performed poorly on languages such as Arabic (ar), Hindi (hi), and Thai (th).

3.2 Discussion

3.2.1 Ablation Study. We performed an ablation study using the official leaderboard set to examine the contribution of the different document representations, as detailed in Table 3. We can see that the model benefits from the combination of embeddings trained with different types of information. The full model improves in both precision and recall compared to the combination of e_{BERT} with any one of the other individual document representations.

3.2.2 Joint Embeddings vs. Output Ensemble. As our pre-trained document embeddings except for e_{KG} are trained on the SHINRA2020-ML task itself, the output of individual sub-models can also be used as a direct prediction output. We consider two different ways of integrating these different types of information. One is *joint embeddings*, which we used for the final submission, i.e., combine the document embeddings as in (1) and train a single classifier to generate the final output. The other is *output ensemble*,

[12] https://www.nlp.ecei.tohoku.ac.jp/projects/AIP-LB/task/shinra2020-ml

	Submission name	Leaderboard $\mathcal{P}$	Leaderboard $\mathcal{R}$	Leaderboard $\mathcal{F}_1$	Test (official evaluation) $\mathcal{P}$	Test (official evaluation) $\mathcal{R}$	Test (official evaluation) $\mathcal{F}_1$	Rank (official)
Ours	`jointrep`	75.3	76.4	75.7	**81.77**	83.71	**82.73**	1
	`jointrepPostprocess`	75.9	77.0	76.3	81.46	83.71	82.57	3
	`jointrepUnionPostprocess`	75.4	78.5	76.4	80.66	84.80	82.68	2
Best-other	BERT (FPTAI)	—	—	—	79.65	**85.00**	82.23	

Table 1: Results (English) on the leaderboard set and the test set (%). "$\mathcal{P}$", "$\mathcal{R}$" and "$\mathcal{F}_1$" denote micro-averaged precision, recall and F1 score, respectively. "Best-other" indicates the result of the best official submission other than ours.

	ar	bg	ca	cs	da	de	en	es	fa	fi	fr	he	hi	hu
Ours	64.55	83.07	**79.82**	81.29	80.56	81.03	**82.73**	**81.39**	80.38	80.91	78.21	81.09	66.67	85.02
Best-other	76.27	83.77	76.28	84.47	82.30	81.86	82.23	80.94	81.70	83.62	81.01	83.79	76.43	85.46
	id	it	ko	nl	no	pl	pt	ro	ru	th	tr	uk	vi	zh
Ours	78.51	**82.02**	82.51	81.64	78.79	84.52	80.87	80.83	82.90	65.02	84.85	81.61	77.06	78.58
Best-other	81.93	81.92	83.67	83.29	80.53	84.53	83.23	84.60	84.08	81.26	86.50	83.12	80.34	81.25

Table 2: Results for all languages we participated in, in terms of micro-averaged F1 score (%). For English, we show the result of the `jointrep` model. "Best-other" indicates the results of the best official submission other than ours.

Model	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$
Full	75.3	76.4	75.7
Full $-e_{\text{KG}}$	74.8	75.8	75.2
Full $-e_{\text{VL}}$	74.5	75.5	74.8
Full $-e_{\text{SS}}$	75.5	77.2	76.0
Full $-e_{\text{CH}}$	74.4	76.4	75.1
$e_{\text{BERT}} \oplus e_{\text{KG}}$	70.8	71.3	71.0
$e_{\text{BERT}} \oplus e_{\text{VL}}$	73.5	74.8	73.9
$e_{\text{BERT}} \oplus e_{\text{SS}}$	70.5	71.6	70.9
$e_{\text{BERT}} \oplus e_{\text{CH}}$	74.2	75.2	74.5

Table 3: Ablation study using different combinations of document representations. The performance is computed on the learderboard data using the raw outputs without any post-processing. "Full $-e_{\text{X}}$" denotes input representation e_{X} being ablated from the input.

Model	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$
Knowledge graph	70.8	71.3	71.0
Text-image	72.2	72.2	72.2
Page layout	61.0	61.1	61.0
Class hierarchy	73.3	75.5	74.0
Majority voting	73.9	75.8	74.5
Joint embeddings	75.3	76.4	75.7

Table 4: Performance of sub-model outputs and majority voting on the leaderboard data.

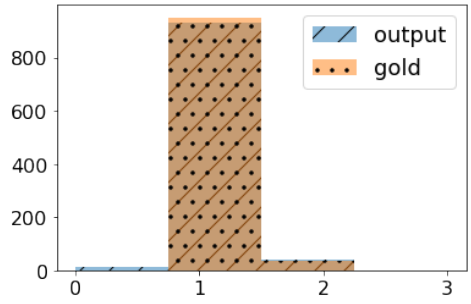

Figure 3: Distribution of the number of predicted and gold-standard ENE labels in our internal development data.

in which we directly use the output of individual sub-models[13] and employ majority voting to aggregate the outputs. Table 4 shows the results on the leaderboard data. Although majority voting improves over the performance of single models, we observe better performance when the joint embedding strategy is used. Therefore, we decided to adopt joint embeddings approach for our final submissions.

3.2.3 Error Analysis. We performed error analysis on our internal development set.[14] Figure 3 shows the distribution of the number of predicted and gold-standard ENE labels per input document. For both the predicted and gold-standard annotation, most articles are

[13] As e_{KG} is not trained on the SHINRA2020-ML task, we use the final model with the document representation $e_{\text{BERT}} \oplus e_{\text{KG}}$ instead.
[14] As we found slight inconsistencies in the training and development splits used for training the sub-models and the final model, we use the intersection of all the development data for error analysis in this section. The size of the dataset is 987 documents.

Page ID	Title	Opening text	Predicted ENEs	Gold ENEs
54866853	Hedi (Policy)	Hedi was an economic policy of the imperial China. It was the state purchase of food supplies from farmers. As a means to control the price of grains and foods, it is an early example of Government procurement. The policy was adopted in the year of 488 by Emperor Xiaowen of Northern Wei as a counter measure of drought. The state purchases food supplies and stock them. ...	`1.7.21.8:Plan`	`0:CONCEPT`
299259	Roman currency	Roman currency for most of Roman history consisted of gold, silver, bronze, orichalcum and copper coinage (see: Roman metallurgy). From its introduction to the Republic, during the third century BC, well into Imperial times, Roman currency saw many changes in form, denomination, and composition. A persistent feature was the inflationary debasement and replacement of coins over the centuries. ...	`1.7.25.1:Currency`	`9:IGNORED`
788538	Azamino	Azamino (あざみ野) is a bedroom community of Tokyo and Yokohama, located in Aoba-ku, Yokohama, Kanagawa Prefecture, Japan. The area is 20 minutes from Shibuya Station on the Tōkyū Den-en-toshi Line and 27 minutes by subway from Yokohama Station on the Yokohama Subway. Known as an upscale enclave, its residents have come to endearingly refer to themselves as Azaminese. It has seen development in recent years. It is located next to the trendy Tama Plaza.	`1.5.0:Location_-Other`	`1.5.1.1:City`
22786903	Hiro H1H	The Hiro H1H (or Navy Type 15) was a 1920s Japanese bomber or reconnaissance biplane flying boat developed from the Felixstowe F.5 by the Hiro Naval Arsenal for the Imperial Japanese Navy. The aircraft were built by Hiro, the Yokosuka Naval Arsenal and Aichi.	`1.7.6:Weapon`, `1.7.17.3:Aircraft`	`1.7.17.3:Aircraft`
586540	Tailor	A tailor is a person who makes, repairs, or alters clothing professionally, especially suits and men's clothing. Although the term dates to the thirteenth century, tailor took on its modern sense in the late eighteenth century, and now refers to makers of men's and women's suits, coats, trousers, and similar garments, usually of wool, linen, or silk. ...	`1.7.23.1:Position_-Vocation`	`0:CONCEPT`

Table 5: Examples of errors from the development set. "Predicted ENEs" shows the prediction of our `jointrep` model.

assigned a single ENE category, and the distributions are almost identical.

Table 5 shows examples of errors from our internal development set. One of the most common types of error is confusion between `CONCEPT` and other classes. Particularly common is confusion between `CONCEPT` and a subclass of `1.7.21:Doctrine_Method`, as shown in the first row in the table, indicating the difficulty of distinguishing these classes. Another frequent type of error is observed between `IGNORED` and others. For example, the article of *Roman currency* is predicted as `Currency` by our model. However, the article does not correspond to a particular currency but describes the history and general characteristics of Roman currency, and thus it should be categorized as `IGNORED`.

Some errors seem to reflect the way the training dataset created, i.e., by propagating labels from Japanese articles using interlanguage links. For example, *Azamino* is predicted as `Location_-Other` while its gold label is `City`. Actually, the English article of *Azamino* is very short compared to the Japanese one, and there is no clue as to its city status. In another example, *Hiro H1H* is described as *a 1920s Japanese bomber or reconnaissance biplane flying boat* in the English article, while it is just mentioned as a flying boat in its corresponding Japanese article. It seems to be reasonable that the model predicts the additional ENE class of `Weapon` based on the English article.

There is another type of error that seems to come from page redirects. In the final example, the English article *Tailor* is predicted to be `Position_Vocation`. However, the corresponding Japanese article is about 洋裁 *(Tailoring)*, and thus the gold label is `CONCEPT`. This happens because the English link for *Tailoring* redirects to the *Tailor* page. These examples suggest the necessity of taking such inconsistency between articles in different languages into consideration for semi-automated construction of structured knowledge bases.

4 CONCLUSIONS

We participated in 28 out of the 30 subtasks (languages) in the SHINRA2020-ML task, with a primary focus on English. Based on different data modalities associated with Wikipedia pages, our system ranked first on four subtasks including English.

Future work includes further investigation of the results with respect to the contribution of each sub-model, as well as applying our fully integrated model to languages other than English to see whether it generalizes well multilingually.

ACKNOWLEDGMENTS

This research was carried out in part using computational resources of AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST).

REFERENCES

[1] [n.d.]. *Extended Named Entity (ver.8.0.0).* http://ene-project.info/ene8/taxonomy/

[2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 6077–6086.

[3] François Chollet. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1800–1807.

[4] 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)*. 4171–4186. https://doi.org/10.18653/v1/N19-1423

[5] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A deep semantic natural language processing platform. arXiv:arXiv:1803.07640

[6] Siddharth Gopal and Yiming Yang. 2013. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 257–265.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778.

[8] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015*. http://arxiv.org/abs/1412.6980

[9] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[10] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael S. Bernstein, and Fei-Fei Li. 2017. Visual Genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision* 123, 1 (2017), 32–73.

[11] Adam Lerer, Ledell Wu, Jiajun Shen, Timothee Lacroix, Luca Wehrstedt, Abhijit Bose, and Alex Peysakhovich. 2019. PyTorch-BigGraph: A large-scale graph embedding system. In *Proceedings of the 2nd SysML Conference*.

[12] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.

[13] Satoshi Sekine, Masako Nomoto, Kouta Nakayama, Asuka Sumida, Koji Matsuda, and Maya Ando. 2020. Overview of SHINRA2020-ML Task. In *Proceedings of the NTCIR-15 Conference*.

[14] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2556–2565.

[15] Aili Shen, Bahar Salehi, Jianzhong Qi, and Timothy Baldwin. 2020. A general approach to multimodal document quality assessment. *Journal of Artificial Intelligence Research* 68 (2020), 607–632.

[16] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations*.

[17] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VL-BERT: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SygXPaEYvH

[18] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception architecture for computer vision. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2818–2826.

[19] Mingxing Tan and Quoc V Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946* (2019).

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[21] Jie Zhou, Chunping Ma, Dingkun Long, Guangwei Xu, Ning Ding, Haoyu Zhang, Pengjun Xie, and Gongshen Liu. 2020. Hierarchy-aware global model for hierarchical text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 1106–1117.

[22] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE International Conference on Computer Vision*. 19–27.