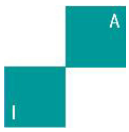
Inteligencia Artificial, 26(72) (2023), 124-136
doi: 10.4114/intartif.vol26iss72pp124-136



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/http://journal.iberamia.org/

Online Incremental Learning Based on Crowdsourcing for Indonesian Ontology Relation Extraction

Eunike Andriani Kardinata[1], Nur Aini Rakhmawati[2]

[1]Department of Information Systems. Institut Sains dan Teknologi Terpadu Surabaya, Indonesia
eunike@istts.ac.id

[2]Department of Information Systems. Institut Teknologi Sepuluh Nopember, Indonesia
nur.aini@is.its.ac.id

Abstract Ontology is a form of structured knowledge representation. Ontology is largely used and developed in the process of information retrieval because of its ability to represent knowledge in a form that is both understandable by machine and human. With the increase of ontology scale and complexity is a greater challenge in extra-logical error identification. Most ontological engineering methods depend on machine learning where there is a risk of overlooking extra-logical error. One way to handle this is by crowdsourcing, that is dividing a large task into several smaller subtasks and employ the mass to complete them online. To utilise crowdsourcing, we change the offline and batch data processing into the online and incremental one. Online incremental learning constructs a model in an iterative manner right after a change is made, ensuring that previously acquired knowledge is maintained. The crowdsourcing participants will be asked to repeatedly validate those relations until the desired accuracy value is reached. From this research, we find that crowdsourcing is able to improve the model used in relation extraction process, from the F1-Score of 87.2 % to 89.8 %. This improvement using crowdsourcing reaches the same score as that using expert. Therefore, crowdsourcing is considered as able to correct extra-logical error accurately, just like expert. Besides, we also discover that offline incremental learning using Random Forest produces a model with higher accuracy than online incremental learning using Mondrian Forest. Random Forest model has the final accuracy value of 90.6 % while Mondrian Forest model has 89.7 %. From this result, we conclude that online incremental learning is unable to produce a better result than offline incremental learning in improving meronymy relation extraction process.

Keywords: Crowdsourcing, Extra-Logical Error, Online Incremental Learning, Relation Extraction.

1. Introduction

The volume of data in the current days is growing very rapidly. A data analyst is required to be able to process data into useful information. Then based on the existing information, we draw a knowledge or an understanding that enables us to make a decision. With the advancements of technology, there are more forms and tools to represent information as well as knowledge.

One of the advancements today is semantic web. Semantic web is an expansion of World Wide Web with the purpose of organising the data in the web into a structure that is readily readable by machine. The standard format in semantic web is established by the World Wide Web Consortium by making use of technologies, such as Resource Description Framework [2] and Web Ontology Language. Ontology is a

ISSN: 1137-3601 (print), 1988-3064 (on-line)

form of knowledge representation that is well-structured. Ontology is a collection of concepts or entities in a domain which are connected by relations between concepts. In simple terms, ontology depicts how objects (entities) within a certain scope (domain) are connected and the types of those connections (relation).

Ontology is widely used and developed because of its ability to define common understanding in a domain, thus making it easier for the use and further development of that knowledge. Bu using ontology, we could know the standard definition of an object and its relation with another object. Besides that, ontology is also able to represent human knowledge into a form that is understandable by machine to be processed further. This is because ontology has a standardised format, just like the format of digital document.

In the development of ontology, we need to know which domain is going to be covered in that ontology, such as food and beverages, health and medicine (medical), and many others. Afterwards, we also decide the purpose of using the ontology; for instance, to assist in information acquisition process, to be used in an application giving recommendations of foods and beverages to its users, or to assist in decision making for medical treatments. From this purpose, ontology developers are able to further define the questions that the ontology should be able to answer. Another equally important consideration is who is going to use and maintain the ontology being developed. From the whole of this process, the ontology developer could decide whether an additional interface is needed to assist users and ontology maintainer.

One area where ontology is used is in information retrieval. Some examples of information retrieval are done in the fields of academic [6], e-commerce [16], and science [13]. It is highly possible that research on open domain could be conducted, so that the results could be applied in general in various fields.

When an ontology is highly used and more new data are added into it, the scale and the complexity of the ontology also rises. With a greater scale and higher complexity, there is an even greater challenge to identify extra-logical error [9]. Extra-logical error is a type of error that is only detectable by human because it is usually related to the context of the sentence.

In linguistics, Avram Noam Chomsky [3] proposed that a sentence consists of deep structure and surface structure. As an example, the sentences "John likes Jane.ªnd "Jane likes John"have different surface structures, but they are both derived from similar deep structure. As such, the two sentences can be considered as having the same meaning.

A sentence may have a logical form representation. Logical form is a representation of human understanding which is solely obtained from the surface structure of the sentence. In other words, in Indonesian language, this form is akin to how we depict the explicit meaning of a sentence. Based on this understanding, a logical error can be defined as an error that occurs at the surface structure of a sentence or due to a mistake in the representation at that structure.

An error that is not logical may be referred as extra-logical or non-logical. Based on their definitions, extra-logical means it is beyond the boundary of logical, whereas non-logical means it is not related to or based on a logical concept. In this research, the type of error being raised is more accurately referred as extra-logical error. This is because the error is still related to the logical form, but in order to discover the true meaning of the sentence, we have to look beyond the boundary of the logical form.

Currently, ontology development methods tend to depend on machine learning methods [7]. Machine learning methods require less resource and time as compared to manual process, but there is a risk of overlooking the extra-logical errors. These overlooked errors affect the quality of the resulting ontology, especially its accuracy. That is why, it is necessary to research on how to handle extra-logical errors to improve the quality of ontology.

Human evaluation may be done by experts or commoners. The advantage of evaluation by experts is that the quality of the evaluation would be higher because experts possess sufficient knowledge of the field. However, the disadvantage is that the cost of employing experts is relatively expensive; usually it is proportional to the quality of resulting evaluation. The other alternative is by utilising the knowledge of the commoners or crowdsourcing. The cost of crowdsourcing is generally less than that for experts. Moreover, there are various established methods that could be used to ensure the validity of mass evaluation.

Some related research have been conducted before. Research about crowdsourcing to tackle extra-logical error had been conducted by Mortensen [9] and Yang [14]. In relation to crowdsourcing, we need incremental learning as researched by Losing [7] and Meng [8]. Other than that, crowdsourcing also requires an interactive medium for its participants. There are several examples of the use of ontology editor

named OntoCop, such as that shown in the research about public comments on government regulations by Yang [15] and in the research about ontology for information science by Sawsaa [12].

As of now, we have yet to find a research that discusses in detail the process of relation extraction for ontology using online incremental learning and crowdsourcing at the validation or data preprocessing stages. Besides, there is much unexplored area in the development of ontology for Indonesian language. With that, we hope that through this research, we could build a highly accurate ontology in Indonesia language.

In this article, we will elaborate on our research in the following manner. Part 2 contains the overview on how we conducted our research. Part 3 elaborates how our experiment was done. After that, we will present the results our our experiments and our analysis on part 4. Finally, we will conclude our research on part 5 and suggest some ideas for future research.

2. Methodology

In their research, Mortensen [9] proposed an overview of framework for crowdsourcing, and Losing [7] also presented an online learning schema which was used in the online incremental experiment. By carefully observing these two frameworks, we designed our system architecture as shown in Figure 1.

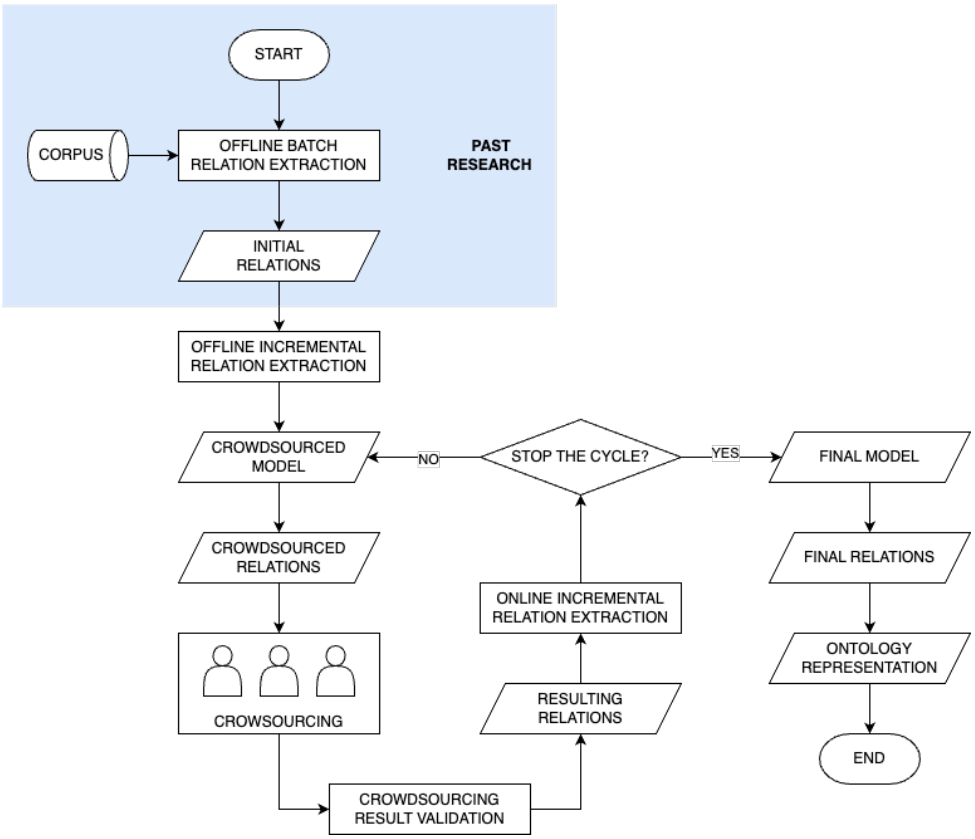


Figura 1: System Architecture

Based on this system, the process began with the offline relation extraction using existing corpus. In the existing corpus, all concepts had been labelled and the initial relations were obtained using offline batch relation extraction method. These stages were not covered in our research; instead, we used the resulting data from the past research by Phi et al. [11]

Afterwards, using the initial relations, we performed offline incremental relation extraction to construct the crowdsourcing model. In this stage, we extracted several features from the existing data. The features

include:

1. Probability of a pattern indicating a relation,
2. Probability of an entity pair indicating a relation,
3. String similarity between entities in a pair (entity matching), and
4. Cosine similarity between entities in a pair (word vector).

The features obtained were trained using Random Forest algorithm to construct our initial model. This model would be used to determine the relations to be corrected by the crowdsourcing participants.

Following, we picked several samples of relations that the crowdsourcing participants would evaluate. The chosen relations were those from the sub-type with the lowest accuracy. These relations would be presented to the participants using a visualisation tool that we have developed. A common and easily understandable representation in this case was a graph.

The crowdsourcing stage would mainly be targeted to involve adults, with the assumption that these participants would be able to comprehend given entities. Nevertheless, the participants did not necessarily need to understand the concept of meronymy relations. This was done in consideration of simulating the crowdsourcing process to be as realistic as possible under the existing limitations. We expected that the majority of crowdsourcing participants would probably have never learnt about the concepts of the relations to be extracted.

The results from the crowdsourcing stage would be validated before they are used in the process of online incremental relation extraction. The validation was done by measuring the inter-annotator agreement [1], that is how much among all of the annotations that they assent with one another. For homogeneous data, the inter-annotator agreement coefficient could be calculated directly. Whereas for heterogeneous data, the inter-annotator agreement coefficient should be calculated based on each collection of homogeneous data, instead of directly calculated from the whole data.

For our research, the measurement of the inter-annotator agreement followed that of Fleiss' Kappa [5]. This measurement was chosen because Fleiss' Kappa could be used when there were more than two annotators. This differed from Cohen's Kappa [4] which was intended for annotation involving just two annotators.

In the first cycle, using valid corrections from the crowdsourcing stage, we would perform the online incremental relation extraction and checked the resulting accuracy. We expected that the accuracy would probably not improve so much just from one round of correction. Thus, in this case, we would incorporate this revision in the crowdsourced model, and repeat the cycle again.

The crowdsourcing cycle would be repeated until it was determined that the model has achieved the target accuracy, hence ending the cycle. There were several possible conditions to stop the cycle. One option was leaving the decision to the participants, that is until they confirmed that the relations no longer needed to be corrected. Another option was to define a value for the target accuracy. Finally, we could also set a fixed number of corrections to be done by the participants.

For each condition to stop the cycle, there exist its own pros and cons. Leaving the decision to the participants was simple, but there was a risk of never-ending cycle because either the participants were never satisfied with the corrections or the model failed to generate good-enough relations. Likewise, setting a definite accuracy value was also simple, but it was rather difficult because we could not know the exact initial accuracy and how much it would improve in the end. Therefore, here we decided to go with the third option, that is deciding a fixed number of corrections to be made. Although we might not get the most optimum number of corrections to be made, at least we would be able to balance between improving the accuracy and ending the cycle before it reached saturation.

As the crowdsourcing cycle ended, we obtained the final model of relation extraction which was used to extract the final relations. These relations would then be visualised as a simple ontology in the form of a graph. We hope that by using the resulting ontology, we could improve the information retrieval process in the open domain.

3. Experiment

This section is organised according to the methodology we explained in the previous section. In more detail, we will elaborate on: data collection, data preprocessing, feature extraction, implementation of offline incremental learning, and implementation of online incremental learning.

3.1. Data Collection

The data used in this research were sourced from previous research titled Ranking-Based Automatic Seed Selection and Noise Reduction for Weakly Supervised Relation Extraction by Phi, et al. [11] The original data were in English and we manually translated them into Indonesian. The data in Indonesian language maintained the same format as the original data.

Based on the format of the original data, we only took some parts of them to be processed in Indonesian language. More specifically, we took the first entity, second entity, pattern showing the relation, source sentence, sub-type of relation, and the order showing which entity is the 'part' and which entity is the 'whole'. We did not translate all data; only those that we were sure would be used were translated. Nevertheless, if needed be, we might translate and release the complete data in Indonesian language or add other data for experiments.

3.2. Data Preprocessing

In this stage, the translated data were prepared into an appropriate format before we extracted the features. There were several steps of preprocessing done, namely: loading the data into the program, adjusting data types for feature extraction, and categorising data to ease the extraction of features. Following, we will provide more details for each step.

3.2.1. Loading Data

The translated data files were available in two formats: .ann and .txt. The files with .ann extension were the annotated data from the past research. At that time, the researcher had used BRAT rapid annotation tool[1]. A sample of .ann file is provided in Figure 2. In each of this file, we have the entity types of both the first and second entities, position indices of both entities in the sentence (starting from 0), first and second entities, relation sub-type, and the part-whole order as explained in 3.1 section.

```
T1      MISC 0 7        Protein
T2      MISC 21 31      asam amino
R1      Component-Of Arg1:T2 Arg2:T1
```

Figura 2: Sample of .ann File

Next, the files with .txt extension contain the source sentence. We loaded the data by utilising common Python libraries, such as glob[2] and os[3]. This process of loading data was closely related to the next process of adjusting the data types.

3.2.2. Adjusting Data Types

We adjusted the data types as we loaded the data into our program. Every file would be read line by line, and for each line, we would acquire the data separated by white-space delimiter. We would check each type of the data and append them into appropriate arrays. Using this method, we also loaded the source sentences into a dedicated array for the whole data. With this, we could easily process the required data by referring to the allocated arrays.

[1]https://brat.nlplab.org/
[2]https://docs.python.org/3/library/glob.html
[3]https://docs.python.org/3/library/os.html

3.2.3. Categorising Data Types

The aim of this step was to group the data which were entity pairs as well as patterns. The collection of entity pairs could be acquired based on the labels found in the sourced data, while the collection of patterns could be acquired by taking the words between the two entities in the source sentences.

From the complete data, we made two more arrays: one was for storing entities only and another for storing patterns only. Generally, we used the same method to create both arrays. The difference was for pattern array, we had to add codes to extract the pattern from the source sentence. In contrast, for entity array, we could just directly take the entities from the sourced data.

As we obtained the entities, we also checked for the direction of the relation. We standardised the format, such that the first entity was the 'part', followed by the second entity which was the 'whole'. In addition, we also assumed that if there were no words between to entities in a sentence, it meant there was no pattern for this entity pair. Finally, for both entity array and pattern array, we made sure that there were no duplicates of data.

3.3. Feature Extraction

The feature extraction process was done to form the initial model to predict the sub-type of relation. Some features used in the learning process were: probability of a pattern indicating a relation, probability of an entity pair indicating a relation, string similarity between entities in a pair, and cosine similarity between entities in a pair.

3.3.1. Probability of a pattern indicating a relation

This feature was obtained by comparing the pattern from the sentence, which the relation was going to be predicted, with the patterns that had been discovered previously. If the pattern in the sentence was found to indicate a certain sub-type of relation, the probability for that sub-type of relation would be 1 (one). If the pattern was never discovered previously, the probability would be 0 (zero). In this process, we defined the result for each pattern by using a vector with the length of eight, signifying the eight relation sub-types. An example for this result is as shown in Figure 3.

```
Pattern : 'adalah bagian dari'
Result  : [ 1       0       1       1       1       1       1       1 ]
```

Figura 3: Sample Result of Pattern Probability

From the given example, the mentioned pattern was found in almost all sub-types of relation, except for one. In the vector, the order of relation sub-types from the first value to the last value follows: Component-Of, Contained-In, Located-In, Member-Of, Participates-In, Involved-In, Subquantity-Of, Constituted-Of. Thus, the mentioned pattern was never used previously for the relation sub-type of Contained-In.

3.3.2. Probability of an entity pair indicating a relation

With similar method as that for the probability of a pattern indicating a relation, we determined the probability of an entity pair indicating a relation. The result from this process was defined just like the probability of a pattern indicating a relation, that was, a vector with the length of eight for eight relation sub-types. The resulting vector also followed the same order of relation sub-types.

3.3.3. String similarity between entities in a pair (entity matching)

The next feature was the similarity between a pair of entities based on the string distance between those two entities. The string distance was calculated using the implementation of Needleman-Wunsch algorithm [10]. First, we created a matrix showing the comparison score for each character in the string of the two entities. Then, from that matrix, we would find a path with the highest score. Based on this

path, we obtained the score for string distance, which could be converted into a value in the range of 0 (zero) up to 1 (one). The score zero meant the entities did not have any similarity at all, while the score one meant the entities were exactly the same.

3.3.4. Cosine similarity between entities in a pair (word vector)

To extract this feature, we represented the entities into vector form and calculated their similarity using cosine similarity. The word embedding used in this research was BERT[4] (Bidirectional Encoder Representations from Transformers) which was published by Google. This embedding was chosen because it was considered to be able to represent contextual information better than other static embedding.

In simple terms, the resulting embedding would have a dimension of 12 layers with 768 values for each layer. From these 12 layers, we would take the values from the last 4 layers because the embedding in these 4 layers were expected to have the highest accuracy. Following, we would elaborate on the process, starting from the embedding to the computation of cosine similarity.

In the beginning, each sentence would go through tokenising process just like other embedding in general. The special tokens used for the embedding in this research were '[CLS]' to show the beginning of the sentence and '[SEP]' to show the end of the sentence. The embedding was done using the previously released model for multilingual embedding, that was, bert-base-multilingual-cased.

The result of the embedding of a sentence was a tensor with the dimension of n 12 768, where n was the number of tokens. Note that a word could be broken down into several tokens, depending on the model being used. Therefore, there was an additional process to obtain the beginning index and the end index of a word. Moreover, an entity could consist of one or more words. Hence, we also added the process to determine the beginning index and the end index of the tokens showing an entity.

For entities consisting of several tokens, we would take the mean value of each token embedding. After that, we would sum the value of the last 4 layers. The result of this process was a vector with the length of 768 elements. This was the vector used to calculate the value of cosine similarity.

3.4. Implementation of Offline Incremental Learning

Based on the system architecture designed, this process aimed to construct the initial model used to generate the relations in the crowdsourcing phase. The implementation of offline incremental learning was done twice using the RapidMiner[5] tool and scikit-learn[6] library in order to ensure the robustness of the model. From this process, we figured the weight of each feature according to their relevance in determining relation sub-types and also the accuracy of the model to be used in crowdsourcing.

In this step, we used the Auto Model feature in RapidMiner. First, we chose the dataset containing all features to be used, and then we decided the task to be done: prediction, clustering, or outlier identification. From the dataset being used, we picked the column being the prediction result. After that, we could also check and manage the target class that we have chosen, followed by choosing the feature data input. Finally, we determined what model was used and how we optimised it, then we execute the model construction.

The construction of model using scikit-learn (sklearn) library was done by preparing the feature data into an appropriate format for training, then executing the functions provided. The raw data of all features would be checked for their completeness and then they were converted into float32 format. From this process, we obtained 5651 rows of data which could be used out of the initial 5727 rows of data. Thereafter, the data would be divided into training and testing set.

The advantage of sklearn library was that we could define the proportion of the data for training and testing, and also determine the random state used to ensure consistent classification results. In this experiment, we used 3:1 ratio for training and testing, and random state of 27.

As the input for model construction, we used all features extracted to produce the output or the target class of eight relation sub-types. We used Random Forest model with the optimisation of a maximum number of 100 trees with depth of 30 for each tree. From this, we acquired the feature weights and the resulting model performance.

[4]https://github.com/google-research/bert
[5]https://rapidminer.com/
[6]https://scikit-learn.org/stable/index.html

There were four features used, namely f1, f2, f3, and f4. Features f1 and f2 were the probability for the eight relation sub-types, thus, there were eight fields for each of them. Feature f3 was the result for the calculation of string distance and the similarity value between a pair of entities. Then feature f4 was the cosine similarity value for a pair of entities.

The weight of each feature showed how relevant a feature was to determine the relation sub-type. It could be seen that each method produced different feature weights, and the accuracy of each method was different as well. The model produced by RapidMiner had the accuracy of 84.3 %, while the model produced by sklearn had the accuracy of 87.3 %. Thenceforth, the model that we would use for crowdsourcing was the model constructed using sklearn library.

According to the sklearn model, we found out that the most relevant feature was the probability of an entity pair indicating a relation with a total weight of 0.399, followed by the probability of a pattern indicating a relation with a total weight of 0.346, the string similarity with a total weight of 0.130, and the cosine similarity with a total weight of 0.125. The weight values were calculated from the average of the values produced by the 100 trees in Random Forest. Hence, there was a chance that the sum of all weight values would not add up to a full value of 1 (one). In this experiment, RapidMiner model lost 0.199 of weight, while sklearn model achieved the full value of 1 (one).

In general, the model constructed had good accuracy. The highest precision score was 99.0 % for relation sub-type Participates-In (showing the relation between the concept and the process in it), and the lowest precision score was 74.8 % for relation sub-type Subquantity-Of (showing the relation between units of matters). The highest recall score was 99.3 % for relation sub-type Located-In (showing the location of entity), and the lowest recall score was 65.5 % for relation sub-type Subquantity-Of.

3.5. Implementation of Online Incremental Learning

The process of implementing online incremental learning comprised the use of the model constructed from the implementation of offline incremental learning, label correction process or crowdsourcing, followed by correction result consolidation and inter-annotator agreement validation, as well as determining whether the correction result would be used to improve the initial model. The experiment done involved three annotators who would check the classification of the eight sub-types of relation.

Based on the implementation of offline incremental learning, we discovered that relation sub-type Subquantity-Of had a low accuracy. Thus, the experiment would be done to improve the model, such that the accuracy for the relation sub-type Subquantity-Of could be increased. In total, there were 536 rows of valid date to be used in this experiment.

The methods to load the data, preprocess the data, and use the model were the same as explained in the previous sections. Furthermore, the process of label correction in the experiment could be done manually, but it could also be done using the interactive tool being developed. The correction results would be consolidated and validated using the calculation of inter-annotator agreement. In this research, we used Fleiss' Kappa score because Fleiss' Kappa considered the number of annotators. This was different from other scores, such as Cohen's Kappa, which was used when the number of annotators were only two.

In the experiment conducted, we calculated the inter-annotator agreement score for the correction results of the relation sub-type predicted by the offline model. The score used was the agreement score for labels in each entity pair. If the agreement score was more than 0.7, we assumed that the correction made by the majority of the annotators was correct. Hence, we would incorporate this correction result to renew the data to be used to improve the online model.

The construction of the online model was done using the scikit-garden[7] library. In this library, there was an implementation of Mondrian Forest, that was, one online version of Random Forest. We used the same configuration to construct the online model, which was a total number of 100 trees with a maximum depth of 30 for each tree. The data used to construct the online model was all valid data of entity pairs and relations (5651 rows of data), whereas for model testing, we only used the data of relation sub-type Subquantity-Of (536 rows of data).

Once the model was constructed and we obtained the new predictions of relations, then the processes of crowdsourcing, correction results consolidation, and updating the model would be repeated in the same

[7]https://scikit-garden.github.io/

way. This iteration would be stopped after a certain number of cycle or when the target accuracy of the model was achieved.

In the next Section 4, we would elaborate more on the resulting accuracy of the online model after employing crowdsourcing. We would also present our analysis and discussion after conducting the experiment.

4. Results and Discussion

After implementing the online incremental learning, here are the results we obtained. The accuracy of the models resulted from online learning was almost the same, but in general, the accuracy of the online model was better than that of the offline model. The following Table 1 shows the accuracy of the model for each cycle of experiment done. There were two accuracy scores calculated, namely the accuracy for all data and the accuracy for the relation sub-type being corrected.

Cuadro 1: Model Accuracy Scores

Model	Overall Accuracy	Sub-type Accuracy
Offline	0.873	0.909
Online (After 1st Correction)	0.886	0.966
Online (After 2nd Correction)	0.895	0.991
Online (After 3rd Correction)	0.893	0.985
Online (After 4th Correction)	0.897	0.994
Online (After 5th Correction)	0.893	0.985
Online (After 6th Correction)	0.897	0.996

The model used in the first correction cycle was the model constructed in the implementation of offline incremental learning with the overall accuracy of 87.3 %. If this model was used for only the relation sub-type of Subquantity-Of, we obtained the accuracy value of 90.9 %. Therefore, in general, we could assume that majority of prediction results were correct, and there should not be too many corrections to be done by the crowdsourcing participants.

After the first correction, the accuracy of the model, as seen on Table 1, increased. The overall accuracy was 88.6 % and the sub-type accuracy was 96.6 %. In the following cycles of correction, the overall accuracy values were about 89 % and the sub-type accuracy values were between 96 % to 99 %.

The overall increase in accuracy values showed that the process of crowdsourcing was useful to improve the accuracy of the model, such that the predicted relations for entity pairs were more accurate. The model's accuracy score after each cycle could be higher or lower. However, those changes were still within a reasonable range, that was, between 0 % to 1 %. This could happen because the corrections made by the crowdsourcing participants might not always be correct, thus, lowering the accuracy of the model.

The final model obtained from the implementation of online incremental learning had the overall accuracy value of 89.7 % and the sub-type accuracy value of 99.6 %. Those scores were the highest among that of all cycles of experiment. With that, we could conclude that the experiment done was able to improve the accuracy of the prediction of relations.

In each cycle of crowdsourcing, we also computed the scores of precision, recall, and F1-measure. The scores are presented in Table 2. We used the weighted scores of precision, recall, and F1-measure for all data. In this calculation, we took the average score of each label by assigning weights according to the number of correct labels. This was different from other calculations, such as macro scores, which the average was directly taken without considering the proportion of each label.

The scores of precision, recall, and F1-measure for online learning in general were within the range of 88 % to 89 %. These scores showed that the performance of the model constructed was quite good, yet the improvement was not very significant. The last updated model had the highest scores of precision, recall, and F1-measure. Therefore, we could conclude that at the end of the online learning process, the model had been successfully improved to be more accurate.

We also computed the scores of precision, recall, and F1-measure in every cycle where the corrections were done by expert. These scores were calculated using weighted method, and we also made sure that the data used in each cycle were the same. The scores are as shown in Table 3. From these results, we discovered that the improvement caused by expert corrections was comparable to that caused by crowdsourcing corrections. Hence, the corrections done by the crowdsourcing participants might be used as an alternative for the corrections by the experts because both had comparably good results.

With regards to the ability of crowdsourcing to correct extra-logical error, we conducted similar experiment, but we did not use online model. In this experiment, we only examined whether the corrections made by crowdsourcing participants could really improve the model's performance.

Using the Random Forest model developed in the beginning of the experiment, the new data from crowdsourcing were able to improve the accuracy of the model. The overall accuracy of the model for all data increased from 87.3 % to 90.6 %, while the sub-type accuracy for Subquantity-Of increased from 90.9 % to 99.4 %. Looking at this improvement, the crowdsourcing process, validated by the computation of inter-annotator agreement, was concluded to be able to improve the model's performance by correcting the data used to construct the model.

Moving on, we compared the accuracy of offline model and online model in the experiment involving crowdsourcing. The initial model used to predict the relations for the first time was the same, that was, the offline model. The following graphic as shown in Figure 4 depicts those scores. Based on the algorithm used, RF stands for Random Forest in offline learning, and MF stands for Mondrian Forest in online learning. The overall accuracy scores are marked as ALL, and the sub-type accuracy scores are marked as SUB.

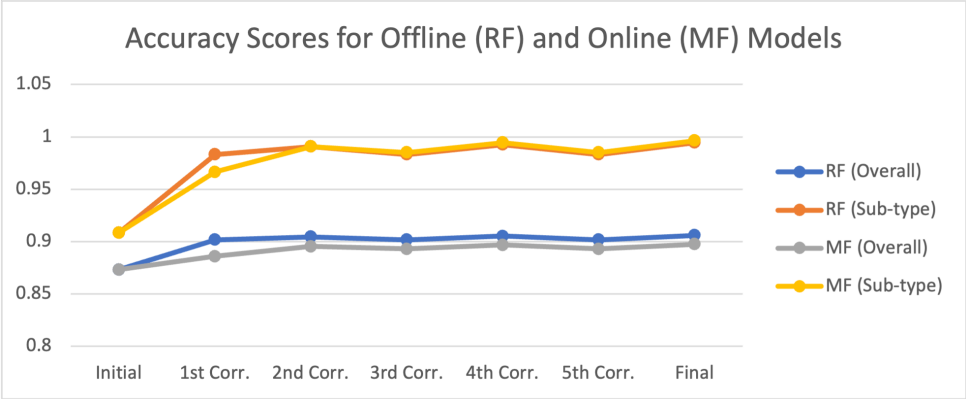


Figura 4: Comparison of Accuracy Scores for Offline (RF) and Online (MF) Models

We discovered that apparently, the overall accuracy of offline model using Random Forest was comparable to that of online model using Mondrian Forest. Nevertheless, the sub-type accuracy of offline model was higher than that of the online model. From this, we concluded that the performance of offline

Cuadro 2: Scores for Precision, Recall, and F1-measure

Model	Precision	Recall	F1-measure
Offline	0.872	0.873	0.872
Online (After 1st Correction)	0.890	0.886	0.887
Online (After 2nd Correction)	0.899	0.895	0.896
Online (After 3rd Correction)	0.897	0.893	0.894
Online (After 4th Correction)	0.901	0.897	0.897
Online (After 5th Correction)	0.897	0.893	0.894
Online (After 6th Correction)	0.902	0.897	0.898

model Random Forest was actually better than the performance of online model Mondrian Forest. From this results as well, we could conclude that based on the accuracy scores, online incremental learning was not able to improve the process of meronymy relation extraction in comparison with offline incremental learning.

5. Conclusions

In this section, we would reiterate our conclusions after conducting the experiments. We would also like to provide some suggestions in order to improve further research in similar field in the future.

The conclusions we drew after finishing our research include the results of our research, the advantages, as well as the disadvantages or the challenges in our research. Following are our conclusions:

1. Based on the experiments done, crowdsourcing was proven to be able to improve the prediction of meronymy relation for a pair of entities by correcting the data used to update the model. The corrections of data from crowdsourcing could be validated using the computation of inter-annotator agreement score, such that the corrections kept were confirmed to be valid.

2. The improvement of the model as a result of corrections made by the participants of crowdsourcing was found to be comparable as that made by the experts. Thenceforth, crowdsourcing might be used as an alternative to validate the results of machine predictions.

3. The performance of offline incremental learning model was apparently slightly better than the performance of online incremental learning model. This was proven by the accuracy of the offline model which was higher than that of the online model.

4. There were two kinds of algorithm for online learning based on Random Forest, namely Online Random Forest[8] and Mondrian Forest[9]. We tried both implementations of these two algorithms, but the latest update was done quite a few years ago. Some dependencies of the Online Random Forest were found to have issues, thus, we continued the experiment using Mondrian Forest through scikit-garden library.

5. The method of online learning could also be defined differently: online in the sense that the model was updated where there was a new feature of the old data, or online in the sense that the model was updated when there was new data. In this research, we were using the second definition. The model used was updated when there were corrections of relation sub-type, or in other words, when there were new data of relation sub-type.

6. In the crowdsourcing phase, the focus of our research was just to correct the relation sub-type without paying attention to the direction of the relation. However, we also found that there were inaccurate relation directions which were not included in the scope of our current research. The direction of relation, indirectly, could also affect the ease of annotators in understanding certain relation sub-type.

[8]https://github.com/amirsaffari/online-random-forests
[9]https://github.com/balajiln/mondrianforest

Cuadro 3: Scores for Precision, Recall, and F1-measure (Expert)

Model	Precision	Recall	F1-measure
Offline	0.872	0.873	0.872
Online (After 1st Correction)	0.890	0.886	0.887
Online (After 2nd Correction)	0.897	0.893	0.894
Online (After 3rd Correction)	0.897	0.893	0.894
Online (After 4th Correction)	0.897	0.893	0.894
Online (After 5th Correction)	0.897	0.893	0.894
Online (After 6th Correction)	0.902	0.897	0.898

7. The challenge faced by the crowdsourcing participants was the difficulty in understanding the difference between each relation sub-type because the concept of meronymy was often considered as just one type. There were risks of incorrect annotations, however we could minimize this by utilising the inter-annotator agreement score.

8. The overall inter-annotator agreement score could be inversely proportional to the individual agreement score for each relation. Therefore, based on what was needed, we could choose which score to be used. In this research, we used the agreement score for each relation because we only took relations with agreement score higher than 0.7 to update the model.

After looking at the disadvantages and the challenges in this research, following are some suggestions to improve the results of future research in similar field. We also include some potential ideas to widen the scope of research which were yet to be included in our research.

1. The next research could be conducted in a fully online setting, which could cover both possibilities of model update. The model could be updated without repeating the whole training process when new features or data are added.

2. The corrections to be done could also include the direction of the relation, such that the resulting ontology, in the form of a directed graph, could be more accurate. This improvement of course would also cover the improvement of the tool used as an interactive medium for crowdsourcing.

3. The validation of corrections could be improved by adding other standards, such that when there are a lot more crowdsourcing participants, we could validate the corrections more carefully. As an example, we could consider to incorporate some weights based on the participant's skill or understanding of the corrections requested. We could judge the participant's ability by giving them some sample data to be corrected beforehand.

4. The experiment could be done using other learning algorithms because each algorithm has its own characteristics which would distinguish its performance in different cases. In this case, the algorithms of Random Forest and Mondrian Forest were found to be sufficiently good for predicting the meronymy relations

We hope that our research would be beneficial for other researchers interested in similar field. We also hope to conduct more experiments to improve our current results and provide a more in-depth analysis in the future.

Acknowledgements

The authors gratefully acknowledge financial support from the Institut Teknologi Sepuluh Nopember for this work under the Publication Writing and IPR Incentive Program (PPHKI) 2023 project scheme.

Referencias

[1] Ron Artstein. Inter-annotator agreement. *Handbook of linguistic annotation*, pages 297–313, 2017.

[2] David Beckett, Tim Berners-Lee, Eric Prud'hommeaux, Gavin Carothers, and Lex Machina. Rdf 1.1 turtle, 2014.

[3] Noam Chomsky. *Frontmatter*, pages 1–4. De Gruyter Mouton, Berlin, Boston, 1957.

[4] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37â–46, 1960.

[5] Joseph L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

[6] Amol N. Jamgade and Shivkumar J. Karale. Ontology based information retrieval system for academic library. In *2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, pages 1–6, 2015.

[7] Viktor Losing, Barbara Hammer, and Heiko Wersing. Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing*, 275:1261–1274, 2018.

[8] Rui Meng, Lei Chen, Yongxin Tong, and Chen Zhang. Knowledge base semantic integration using crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering*, 29(5):1087–1100, 2017.

[9] Jonathan M. Mortensen. Crowdsourcing ontology verification. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul Groth, Chris Biemann, Josiane Xavier Parreira, Lora Aroyo, Natasha Noy, Chris Welty, and Krzysztof Janowicz, editors, *The Semantic Web – ISWC 2013*, pages 448–455, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[10] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

[11] Van-Thuy Phi, Joan Santoso, Masashi Shimbo, and Yuji Matsumoto. Ranking-based automatic seed selection and noise reduction for weakly supervised relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 89–95, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[12] Ahlam Sawsaa and Zhongyu Joan Lu. Virtual community of practice ontocop: Towards a new model of information science ontology iso. *Int. J. Inf. Retr. Res.*, 1(2):55â78, apr 2011.

[13] Liu Xinhua, Zhang Xutang, and Li zhongkai. A domain ontology-based information retrieval approach for technique preparation. *Physics Procedia*, 25:1582–1588, 2012. International Conference on Solid State Devices and Materials Science, April 1-2, 2012, Macao.

[14] Hui Yang and Jamie Callan. Human-guided ontology learning. 01 2008.

[15] Hui Yang and Jamie Callan. Ontocop: Constructing ontologies for public comments. *IEEE Intelligent Systems*, 24:70–75, 2009.

[16] Teng yang TAO and Ming ZHAO. An ontology-based information retrieval model for vegetables e-commerce. *Journal of Integrative Agriculture*, 11(5):800–807, 2012.