NLPatVCU CLEF 2020 ChEMU Shared Task System Description

Darshini Mahendran, Gabrielle Gurdin, Nastassja Lewinski, Christina Tang, and Bridget T. McInnes

Virginia Commonwealth University, Richmond VA 23220, USA
{mahendrand,gurding,nalewinski,ctang2,btmcinnes}@vcu.edu

Abstract. This paper describes our team's participation in the Tracks 1 & 2 from Conference and Labs of the Evaluation Forum (CLEF 2020) Challenge organized by Cheminformatics Elsevier Melbourne University for extracting information over chemical reactions from patents. We discuss our systems: MedaCy, a python-based supervised multi-class entity recognition system, and RelEx, a python-based relation extraction system which includes rule-based and supervised learning pipelines. Our best model for Task 1 obtained an overall relaxed precision of 0.95 and exact precision of 0.87; relaxed recall of 0.99 and exact recall of 0.86; and relaxed F_1 score of 0.97 and exact F_1 score of 0.87. Our best model for Task 2 obtained an overall precision of 0.80; recall of 0.54; and F_1 score of 0.65.

Keywords: Named Entity Recognition (NER) · Relation Extraction (RE) · Event Extraction (EE)

1 Introduction

Chemical Patents are a primary source for information about novel chemicals and chemical reactions. With the increasing volume of such patents, the dissemination of information about these chemicals and chemical reactions has become even more labor and time intensive. This information can be used to discover new chemicals and synthetic pathways[1][11]. Therefore, informatics tools for automatically extracting information from these documents are more important than ever.

The process of extracting relevant information from chemical patents has been referred to as chemical reaction detection [12], and two of the main steps in this process are identifying the different parts of a chemical reaction within these documents and then identifying the relationships between them. This can be accomplished with Named Entity Recognition (NER) – the automatic labeling of certain spans within text corresponding to specific labels; and Event Extraction

(EE) – the automatic classifying and linking entities based on their relationships to each other.

The CLEF 2020 ChEMU [7] Task 1 aims to create systems to perform NER over chemical patents as the first step in chemical reaction detection. Specifically, the goal of this task is to automatically identify chemical compounds based on the role they play in a reaction, as well as other relevant information such as yield and temperature. The CLEF 2020 ChEMU Task 2 aims to create systems to perform EE over the entities to identify the individual steps in the reaction.

In this paper, we describe our participation in the CLEF 2020 ChEMU Task 1 and Task 2 Challenge. For this challenge, we used our python framework MedaCy [1] to automatically identify the experimental parameters associated with the reaction including the trigger words used to link the parameters; and RelEx [2] to automatically link the trigger words with the experimental parameters to provide the sequence of steps within the reaction. MedaCy contains a number of supervised multi-label sequence classification algorithms for NER. RelEx contains a rule-based and supervised learning-based algorithms to identify relations between entities. Our best models for Task 1 obtained an overall relaxed precision of 0.95 and exact precision of 0.87; relaxed recall of 0.99 and exact recall of 0.86; and relaxed F_1 score of 0.97 and exact F_1 score of 0.87. Our best model for Task 2 obtained an overall precision of 0.80; recall of 0.54; and F_1 score of 0.65.

Table 1: Entity type statistics of the dataset

Entity Type	**Definition**
REACTION_PRODUCT (R.P.)	A product is a substance that is formed during a chemical reaction.
STARTING_MATERIAL (S.M.)	A substance that is consumed in the course of a chemical reaction providing atoms to products is considered as starting material.
REAGENT_CATALYST (R.C.)	A reagent is a compound added to a system to cause or help with a chemical reaction. Compounds like catalysts, bases to remove protons or acids to add protons must be also annotated with this tag.
SOLVENT (S)	A solvent is a chemical entity that dissolves a solute resulting in a solution.
OTHER_COMPOUND (O.C.)	Other chemical compounds that are not the products, starting materials, reagents, catalysts and solvents.
TIME	The reaction time of the reaction.
TEMPERATURE (Temp)	The temperature of the reaction.
YIELD_PERCENT (Y.P.)	Yields given in percent values.
YIELD_OTHER (Y.O.)	Yields provided in other units than %.

2 Data

The CLEF 2020 data corpus [7] includes chemical entities and events that explain the sequence of steps that leads a chemical reaction to an end product. It

[1] https://github.com/NLPatVCU/medaCy/

[2] https://github.com/NLPatVCU/RelEx/tree/CLEF_2020

includes 10 different entity labels described as shown in the Table 1. The ARG1 event label corresponds to relations between a trigger word (REACTION_STEP, WORKUP) and chemical compound entities. Table 2 shows the event statistics of the training dataset. The ARGM event label corresponds to the relations between a trigger word and temperature, time, or yield entities.

Table 2: Number of entity types and trigger words in the training data and their event relations

Events	**Entities**	**Instances**	**REACTION_STEP**	**WORKUP**
ARG1	EXAMPLE_LABEL	886	-	-
	REACTION_PRODUCT	2052	1101	11
	STARTING_MATERIAL	1754	1747	4
	REAGENT_CATALYST	1281	1272	-
	SOLVENT	1140	1134	4
	OTHER_COMPOUND	4640	161	4097
ARGM	YIELD_PERCENT	955	937	1
	YIELD_OTHER	1061	1043	2
	TIME	1059	839	81
	TEMPERATURE	1515	813	242
Triggers	REACTION_STEP	3815		
	WORKUP	3053		

3 Methods

This section describes the underlying methodology of our system.

3.1 Named Entity Recognition and Trigger Detection

To identify the experimental parameters and triggers from the data, we use MedaCy's bidirectional Long Short Term Memory (LSTM) units with a Conditional Random Field (CRF) output layer implemented in PyTorch [9]. LSTMs [4] are a type of recurrent neural network. They take the current input example as well as what they have seen in the past as their input. Hence, they have two sources of input: their current state and their past states. This allows them to connect previous observations, such as words in a sentence, and learn dependencies of these words over arbitrarily long distances. They incorporate the functionality to identify what information that should be passed to the next component and what information should not, allowing for only relevant information to be passed on. For bi-directional LSTMs (biLSTMs), data are processed in both directions with two separate hidden layers, which are then fed forward into the same output layer. This allows the system to exploit context in both directions. A linear-chain CRF is used to assign the final class probability. CRFs are a sequence learning algorithm which incorporate the interdependence between labels into model induction and prediction. Therefore, using a CRF output allows the model to use the preceding label predictions to inform what labels are most likely to follow or to occur close together.

The input to our biLSTM+CRF model in this work is pre-trained word embeddings [6] in combination with character embeddings [3]. These embeddings are concatenated and then passed through the network.

The word2vec [6] embeddings are derived from a neural network that learns a representation of a word-word co-occurrence matrix. The character embeddings are learned using a biLSTM and concatenated onto the word2vec embeddings. Fig 1 shows a simple example for the term *mice*. This network is valuable for providing input especially in the case of out-of-vocabulary words. In the case of chemical patents, many tokens are long chemical names that do not show up in the dataset used to train word embeddings, such as, the reaction product *3-Isobutyl-5-methyl-1-(oxetan-2-ylmethyl)-6-[(2-oxoimidazolidin-1-yl)methyl]thieno[2,3-d]pyrimidine-2,4(1H,3H)-dione*.

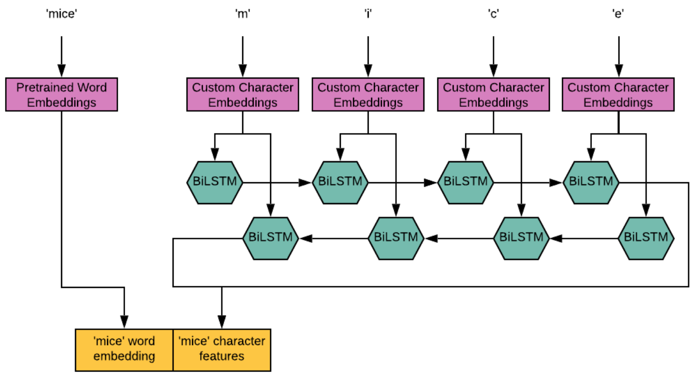
'mice'
'm'
'i'
'c'
'e'
Pretrained Word Embeddings
Custom Character Embeddings
Custom Character Embeddings
Custom Character Embeddings
Custom Character Embeddings
BiLSTM
BiLSTM
BiLSTM
BiLSTM
BiLSTM
BiLSTM
BiLSTM
BiLSTM
'mice' word embedding
'mice' character features

Fig. 1: An illustration of how Character Embedding works

3.2 Task 2: Event Extraction

To identify the trigger words, we use our NER system medaCy as described above. To identify the chemical arguments between the trigger words and the entities, we use RelEx, a python-based Relation Extraction Framework developed to identify relations between two entities. The framework contains two main components: 1) Rule-based Method and 2) Convolutional Neural Network(CNN)-based Method. In this section, we provide a brief overview of each component.

Rule-based Method. RelEx's rule-based method utilizes the co-location information of the trigger words to determine that, with respect to the entity if the word is referring to the trigger word or not. We use a breadth-first search algorithm to find the closest occurrence of the trigger word on either side of the entity and all the closest occurrences of the trigger words within a sentence. For each entity in the data set, we traverse both sides until the closest occurrence of the trigger word is found using the provided span values of the entities. We apply different traversal techniques and determine the best traversal technique. The following are the different traversal techniques we use: traverse left-only, traverse right-only, traverse left-first-then-right, and vice versa. In this work, we use left-only traversal where we traverse to the left side of the entity mention finding the closest occurrence of the trigger words.

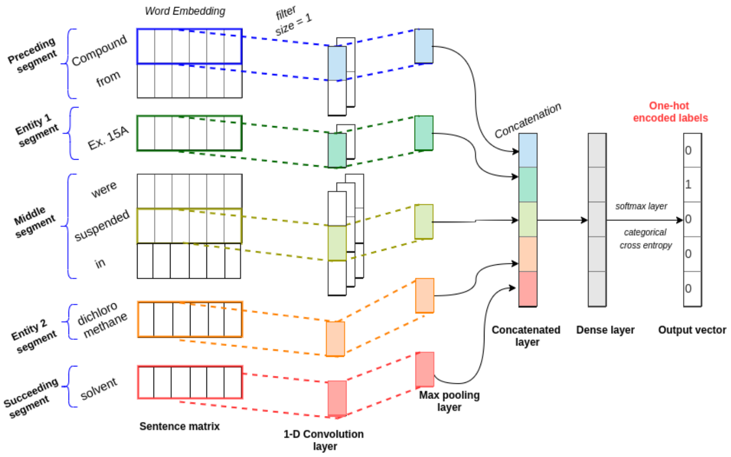

Fig. 2: An illustration of our model for CNN-based method

CNN-based Method. RelEx's CNN-based method automatically extracts and classifies the events. CNNs are a form of deep neural networks and mostly consist of four main layers [8]: embedding, convolution, pooling and feed-forward layers. CNNs allow word embeddings to train on the input text itself or use pre-trained word vectors obtained from an external resource. Initially the convolution layer which is a filter learns using the backpropagation algorithm and extracts features from the input. Then the maxpooling layer uses the position information and helps to extract the most significant feature from the output of the convolution filter. Finally the feed-forward layer uses a softmax classifier that performs classification.

In this work, for each *Trigger word-Entity pair* we perform a binary classification to identify whether there is a relation between the trigger word and the entity or not. First, we identify and extract the sentence where a *Trigger word-Entity pair* pair lies and based on where the text spans are located in the sentence, we divide the sentence into segments as follows:

- preceding - tokenized words before the first concept
- concept 1 - tokenized words in the first concept
- middle - tokenized words between the 2 concepts
- concept 2 - tokenized words in the second concept

– succeeding - tokenized words after the second concept

Figure 2 shows an abstract view of the construction of the CNN-based model. A segment is represented by a matrix of $k * N$ where k is the dimension of the word embeddings and N is the number of words in a segment. In this work, we use ChemPatent pre-trained word embeddings. We construct separate convolution units for each segment and concatenate before the fixed-length vector is fed to the dense layer that performs the classification. Each convolution unit applies a sliding window that processes the segment and feeds the output to the max-pooling layer to extract important features independent of their location. The output features of the max-pooling layer of each segment are then flattened and concatenated into a vector before feeding it into the fully connected feedforward layer. The vector is finally fed into a softmax layer to perform the binary classification whether the relationship exists or not.

3.3 Experimental Details

Word Embeddings . We explore two pre-trained word embeddings: 1) ChemPatent embeddings [7] trained over a collection of 84,076 full patent documents (1B tokens); and 2) WikiPubmed embeddings [10] in our methods.

MedaCy . We used PyTorch [9] for the implementation of the BiLSTM+CRF model. Models were trained for 40 epochs, and optimized using stochastic gradient descent. A window size of 0 generated the best results. Tokenization was conducted using the SpaCy tokenizer. The labels are strictly the entity types.

RelEx . We used Keras [2] for the implementation of the CNN architecture. We experimented with different sliding window sizes, filter sizes, loss functions for fine-tuning and in this work, small filter sizes generated best results for small filter sizes. We applied the dropout technique on the output of the convolution layer to regularize the model. We used *Adam* and *rmsprop* optimizers to minimize our loss function. We trained the models for 5 -10 epochs to avoid over-fitting.

3.4 Evaluation

For Tasks 1 and 2, we report the precision, recall, and F_1 scores. Precision is the ratio between correctly predicted mentions over the total set of predicted mentions for a specific entity; recall is the ratio of correctly predicted mentions over the actual number of mentions, and F_1 is the harmonic mean between precision and recall. For Task 1, we report both the exact and relaxed results for each entity category. In exact evaluation, two annotations are equal only if they have the same tag with exactly matching spans. With the relaxed evaluation, two annotations are equal if they share the same tag and their spans overlap with each other.

4 Results and Discussion

In this section, we discuss the results for Task 1 and 2.

4.1 Task 1: Named Entity Recognition

Results. Tables 3 - 5 show the exact and relaxed precision, recall, and F_1 scores obtained over the testing set for identifying the named entities in each of our three runs. Run 1 model was trained over the training data using the biLSTM+CRF with the CheMU Patent embeddings; run 2 model was trained over the training data using the biLSTM+CRF with the WikiPubmed embeddings; and run 3 model was trained over the training and development data combined with the biLSTM+CRF using the WikiPubmed embeddings. Table 6 shows the baseline results using the CRF-based NER system BANNER [5] provided by the organizers and the overall results of each of our runs.

Table 3: Run 1: Precision (P), Recall (R), and F_1 results using biLSTM+CRF trained over training data With CheMU patent embeddings

	Exact			**Relaxed**		
Entity	**P**	**R**	F_1	**P**	**R**	F_1
EXAMPLE_LABEL	0.94	0.95	0.94	0.94	0.98	0.96
OTHER_COMPOUND	0.9	0.82	0.86	0.97	0.99	0.98
REACTION_PRODUCT	0.84	0.83	0.83	0.9	0.97	0.94
REAGENT_CATALYST	0.85	0.9	0.87	0.88	0.99	0.93
SOLVENT	0.91	0.94	0.93	0.92	1	0.96
STARTING_MATERIAL	0.85	0.84	0.85	0.91	1	0.95
TEMPERATURE	0.63	0.63	0.63	0.99	0.99	0.99
TIME	0.88	0.88	0.88	1	1	1
YIELD_OTHER	0.95	0.98	0.97	0.96	1	0.98
YIELD_PERCENT	0.99	0.99	0.99	1	1	1
System	**0.87**	**0.85**	**0.86**	**0.95**	**0.99**	**0.97**

Overall, the biLSTM+CRF model trained using patent embeddings returned the best results, obtaining a 96.78% system-wide relaxed F_1 score. This model performed better than baseline for all entity labels except EXAMPLE_LABEL, for which it performed almost identically. This model's performance is likely due to the domain-relevant information contained within the embeddings. The best performance for exact evaluation resulted from the model trained over a combination of the training and development sets. However, this model's overall performance was worse than the baseline model. Still, we believe this model's better performance compared to the other models may be due to the increase of volume of data used to train by the addition of the development set.

Although the exact results for the models performed slightly worse than baseline, each of the models performed better on the relaxed results, with the model trained over patent embeddings performing best. This discrepancy may be due to the way that MedaCy handles entity classification. Within MedaCy, each

Table 4: Run 2: Precision (P), Recall (R), and F_1 results using biLSTM+CRF trained over training data with WikiPubmed embeddings

	Exact			Relaxed		
Entity	**P**	**R**	F_1	**P**	**R**	F_1
EXAMPLE_LABEL	0.98	0.93	0.95	0.98	0.98	0.96
OTHER_COMPOUND	0.89	0.84	0.87	0.95	0.98	0.96
REACTION_PRODUCT	0.83	0.82	0.82	0.9	0.97	0.94
REAGENT_CATALYST	0.86	0.89	0.87	0.89	1	0.43
SOLVENT	0.94	0.91	0.93	0.95	0.99	0.97
STARTING_MATERIAL	0.85	0.83	0.84	0.91	0.99	0.95
TEMPERATURE	0.63	0.63	0.63	0.99	0.99	0.99
TIME	0.88	0.87	0.87	1	0.99	1
YIELD_OTHER	0.97	0.98	0.97	0.98	0.98	0.98
YIELD_PERCENT	1	0.99	0.99	1	0.99	0.99
System	**0.87**	**0.85**	**0.86**	**0.95**	**0.98**	**0.96**

Table 5: Run 3: Precision (P), Recall (R), and F_1 results using biLSTM+CRF trained over training and development data with WikiPubmed embeddings

	Exact			Relaxed		
Entity	**P**	**R**	F_1	**P**	**R**	F_1
EXAMPLE_LABEL	0.96	0.94	0.95	0.95	0.96	0.95
OTHER_COMPOUND	0.9	0.84	0.87	0.96	0.98	0.97
REACTION_PRODUCT	0.8	0.82	0.81	0.88	0.98	0.93
REAGENT_CATALYST	0.9	0.88.	0.89	0.93	0.99	0.96
SOLVENT	0.94	0.93	0.94	0.94	0.99	0.96
STARTING_MATERIAL	0.88	0.86	0.87	0.92	0.99	0.95
TEMPERATURE	0.63	0.63	0.63	0.99	0.99	0.99
TIME	0.88	0.88	0.88	1	1	1
YIELD_OTHER	0.98	0.98	0.98	0.98	0.99	0.98
YIELD_PERCENT	0.99.	0.99	0.99	0.99	0.99	0.99
System	**0.87**	**0.86**	**0.87**	**0.95**	**0.98**	**0.97**

individual token is given its own label (‘O’ for unlabelled entities), so for entities with spans long than one token, the entity may have only been partially labelled. For instance, in many cases of the TEMPERATURE label, MedaCy labeled ‘C’ or ‘°C,’ excluding the number preceding the temperature symbol. This may also account for why each model performed poorly for the TEMPERATURE label when evaluating in exact mode, but performed well when evaluating in relaxed mode.

Error Analysis. Confusion matrices for the three runs over the testing dataset are shown in Figures 3 - 5. Rows in the matrix represent annotated entities and columns represent predicted entities. For instance, in 3, YIELD_OTHER (Y.O) was misidentified as YIELD_PERCENT (Y.P.) 28 times. Table 7 shows the acronym of each of the labels used in the confusion matrices. The colors in the matrix indicate the density of the entities and the system annotations. The bottom right corner of each matrix is darker because of the large number of OTHER_COMPOUND (O.C) entities in the dataset.

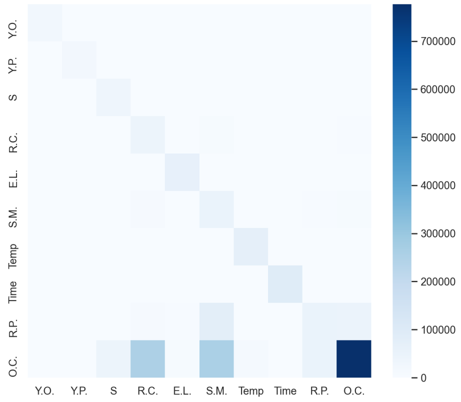

Fig. 3: Run 1 Confusion Matrix using biLSTM+CRF trained over training data with CheMU patent embeddings

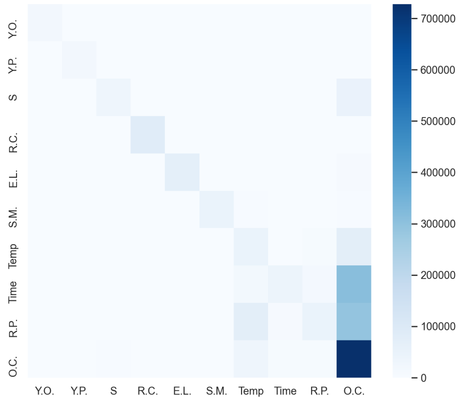

Fig. 4: Run 2 Confusion Matrix using biLSTM+CRF trained over training data with WikiPubmed embeddings

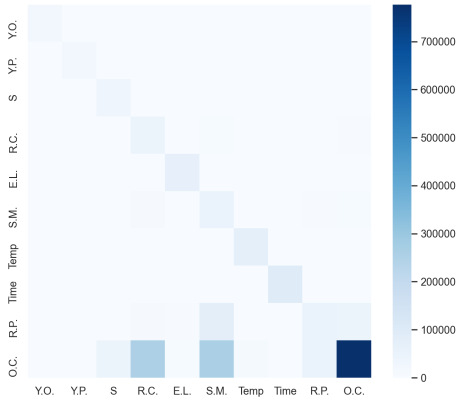

Fig. 5: Run 3 Confusion Matrix using biLSTM+CRF trained over training + development data with WikiPubmed embeddings

Table 6: Task 1 Baseline Results

	Exact			**Relax**		
	P	**R**	F_1	**P**	**R**	F_1
Run 1	0.87	0.85	0.86	**0.95**	**0.99**	**0.97**
Run 2	0.87	0.85	0.86	**0.95**	0.98	0.96
Run 3	0.87	0.85	0.87	**0.95**	0.98	**0.97**
Baseline	**0.91**	**0.87**	**0.89**	0.92	0.95	0.94

Table 7: Key for the confusion matrix figurse

Label	**Acronym**	**Label**	**Acronym**
EXAMPLE_LABEL	E.L.	REACTION_PRODUCT	R.P.
STARTING_MATERIAL	S.M.	REAGENT_CATALYST	R.C.
SOLVENT	S	OTHER_COMPOUND	O.C.
YIELD_PERCENT	Y.P.	YIELD_OTHER	Y.O.
TIME	Time	TEMPERATURE	Temp

The majority of mislabeling occurred when more specific entity labels, such as STARTING_MATERIAL (S.M.), REAGENT_CATALYST (R.C.), or REACTION_PRODUCT (R.P.), were predicted to be OTHER_COMPOUND (O.C.). This may be because the models were able to predict that certain spans contained chemical named, but were too general and unable to predict the specific label. Additionally, spans annotated as OTHER_COMPOUND (O.C.) were consistently predicted to be more specific types of compounds. It seems that while the models are able to predict which spans contain chemical compounds, they are less able to distinguish between the types of compounds.

4.2 Task 2: Event Extraction

Results. Tables 8 - 10 show the exact match precision, recall, and F_1 scores obtained over the testing set for each of our three runs. Run 1 used our RelEx's CNN-based system trained over the ChemPatent embeddings with the trigger words identified using medaCy's biLSTM+CRF trained over the ChemPatent embeddings. Run 2 used our RelEx's rule-based system with the trigger words identified using medaCy's biLSTM+CRF trained with ChemPatent embeddings. Run 3 used our rule-based system with the trigger words identified using medaCy's biLSTM+CRF trained with WikiPubmed embeddings. Table 11 shows the comparison with the co-occurrence baseline provided by the organizers of the ChEMU challenge and the overall results from each of our runs.

The overall results show that all three runs obtain a higher precision and F_1 score than the baseline but not recall. The system results show that the CNN-based (Run 1) model obtains a higher overall F_1 score than both the rule-based (Run 2 & 3) models. When training with CNN the overall precision of the predictions is high but the recall is low, this shows that CNN failed to classify all instances but was able to classify most of the predicted instances correct. Also, we can see the performance of each event class (*Trigger word-Entity pair*) in Run 1 is proportional to the number of instances in the training set. For example, event classes, REACTION_STEP-REAGENT_CATALYST

Table 8: Run1: Precision (P), Recall (R) and F_1 results using CNN-based system with trigger words identified using medaCy trained with CheMU patent embeddings

Argument	**Trigger**	**Entity**	# Train	P	R	F_1
ARG1	REACTION_STEP	OTHER_COMPOUND	161	0.00	0.00	0.00
		REACTION_PRODUCT	1101	0.92	0.96	0.94
		REAGENT_CATALYST	1272	0.78	0.69	0.74
		SOLVENT	1134	0.64	0.74	0.69
		STARTING_MATERIAL	1747	0.82	0.43	0.56
	WORKUP	OTHER_COMPOUND	4097	0.73	0.29	0.42
		REACTION_PRODUCT	11	0.00	0.00	0.00
		SOLVENT	4	0.00	0.00	0.00
		STARTING_MATERIAL	4	0.00	0.00	0.00
ARGM	REACTION_STEP	TEMPERATURE	813	0.83	0.30	0.44
		TIME	839	0.78	0.73	0.75
		YIELD_OTHER	1043	0.93	0.96	0.95
		YIELD_PERCENT	937	0.91	0.94	0.92
	WORKUP	TEMPERATURE	242	0.56	0.08	0.14
		TIME	81	0 .00	0.00	0.00
System				0.81	0.54	0.65

Table 9: Run 2: Precision (P), Recall (R) and F_1 results using rule-based system with trigger words identified using medaCy trained with CheMU patent embeddings

Argument	**Trigger**	**Entity**	# Train	P	R	F_1
ARG1	REACTION_STEP	OTHER_COMPOUND	161	0.02	0.63	0.04
		REACTION_PRODUCT	1101	0.82	0.78	0.80
		REAGENT_CATALYST	1272	0.52	0.35	0.42
		SOLVENT	1134	0.81	0.55	0.65
		STARTING_MATERIAL	1747	0.63	0.31	0.41
	WORKUP	OTHER_COMPOUND	4097	0.90	0.86	0.88
		REACTION_PRODUCT	11	0.01	1.00	0.02
		REAGENT_CATALYST	-	0.00	0.00	0.00
		SOLVENT	4	0.07	1.00	0.14
		STARTING_MATERIAL	4	0.04	1.00	0.08
ARGM	REACTION_STEP	TEMPERATURE	813	0.77	0.89	0.83
		TIME	839	0.85	0.93	0.89
		YIELD_OTHER	1043	0.83	0.80	0.81
		YIELD_PERCENT	937	0.86	0.85	0.85
	WORKUP	TEMPERATURE	242	0.66	0.81	0.73
		TIME	81	0.36	0.53	0.43
		YIELD_OTHER	2	0.00	0.00	0.00
		YIELD_PERCENT	1	0.00	0.00	0.00
System				**0.51**	**0.72**	**0.60**

and REACTION_STEP-STARTING_MATERIAL, have more training instances and obtain a high F_1 score, whereas the event classes, WORKUP-SOLVENT and WORKUP-STARTING_MATERIAL, have a very few instances and obtain an F_1 score of zero.

The rule-based models (Run 2 & 3) obtain comparatively high recall and low precision. The rule-based methods predicts all the closest occurrences of the trigger words of the entity compounds in the traversal area, however many

Table 10: Run 3: Precision (P), Recall (R) and F_1 results using rule-based system with trigger words identified using medaCy trained with WikiPubmed embeddings

Argument	**Trigger**	**Entity**	# Train	P	R	F_1
ARG1	REACTION_STEP	OTHER_COMPOUND	161	0.02	0.63	0.04
		REACTION_PRODUCT	1101	0.82	0.78	0.80
		REAGENT_CATALYST	1272	0.52	0.35	0.42
		SOLVENT	1134	0.81	0.54	0.65
		STARTING_MATERIAL	1747	0.62	0.30	0.40
	WORKUP	OTHER_COMPOUND	4097	0.90	0.86	0.88
		REACTION_PRODUCT	11	0.01	1.00	0.02
		REAGENT_CATALYST	-	0.00	0.00	0.00
		SOLVENT	4	0.07	1.00	0.13
		STARTING_MATERIAL	4	0.03	1.00	0.07
ARGM	REACTION_STEP	TEMPERATURE	813	0.85	0.89	0.82
		TIME	839	0.78	0.93	0.89
		YIELD_OTHER	1043	0.82	0.80	0.81
		YIELD_PERCENT	937	0.86	0.85	0.85
	WORKUP	TEMPERATURE	242	0.61	0.85	0.71
		TIME	81	0.36	0.60	0.45
		YIELD_OTHER	2	0.00	0.00	0.00
		YIELD_PERCENT	1	0.00	0.00	0.00
System				**0.51**	**0.71**	**0.59**

Table 11: Task 2 Baseline evaluation

	P	**R**	F_1
Run 1	**0.81**	0.54	**0.65**
Run 2	0.51	0.72	0.60
Run 3	0.51	0.71	0.59
Baseline	0.38	**0.89**	0.38

predictions are false positives. Since the number of instances in the training set does not affect the rule-based methods, the performance of the event classes that have few instances performs better. For example, the event classes, WORKUP-TIME and REACTION_STEP-OTHER_COMPOUND, obtained zero F_1 score with CNN-based model but performed better with the rule-based models obtaining F_1 scores of 0.43 and 0.88, respectively.

Table 12 shows the arithmetic mean and weighted arithmetic mean of the precision, recall, and F_1 score for both trigger word classes for each run. Bold terms indicate the best performance for each trigger word. We can see the CNN-based method (Run 1) performs well with the REACTION_STEP classes and poor with WORKUP classes. This is because most of the REACTION_STEP classes have more instances for the CNN to train on but most of the WORKUP classes have few instances. This is the same reason the rule-based methods (Run 2 & 3) perform better with those classes. The weighted arithmetic mean results contradict with the arithmetic mean results, as we can see a notable difference in the F_1 score when comparing the classes of REACTION_STEP and WORKUP. The WORKUP event class obtains a better performance due to the significant imbalance between the individual event classes. The weighted arithmetic mean

Table 12: Arithmetic and Weighted arithmetic mean of the performance of the trigger words for each run

Trigger	Entity	Arithmetic mean			Weighted arithmetic mean		
		P	**R**	F_1	**P**	**R**	F_1
REACTION_STEP	Run 1	**0.73**	0.64	**0.67**	**0.81**	**0.69**	**0.73**
	Run 2	0.68	**0.68**	0.63	0.73	0.63	0.66
	Run 3	0.68	0.67	0.63	0.73	0.63	0.65
WORKUP	Run 1	0.14	0.04	0.06	0.70	0.28	0.40
	Run 2	**0.23**	0.58	**0.25**	**0.87**	**0.85**	**0.86**
	Run 3	0.22	**0.59**	**0.25**	**0.87**	**0.85**	**0.86**

allocates more weight to the classes that have more instances and vice versa we see an improvement in the performance of both classes.

Error Analysis. Tables 13 and 14 show a detailed error analysis of the CNN-based (Run 1) and the rule-based method (Run 2) respectively where the trigger words are trained with ChemPatent embeddings. Here we report the number of true positives (tp), false positives (fp), and false negatives (fn) and also ”fpm” and ”fnm”, two metrics that represent the number of false positives and false negatives, of which the corresponding entities are missing.

Table 13: Error analysis for the CNN model trained with ChemPatent embeddings

Argument	Trigger	Entity	tp	fp	fn	fpm	fnm
ARG1	REACTION_STEP	OTHER_COMPOUND	0	0	63	0	11
		REACTION_PRODUCT	436	36	16	11	3
		REAGENT_CATALYST	350	97	155	17	8
		SOLVENT	316	179	111	16	7
		STARTING_MATERIAL	305	68	406	12	9
	WORKUP	OTHER_COMPOUND	516	192	1234	23	73
		REACTION_PRODUCT	0	0	4	0	0
		REAGENT_CATALYST	-	-	-	-	-
		SOLVENT	0	0	2	0	0
		STARTING_MATERIAL	0	0	1	0	0
ARGM	REACTION_STEP	TEMPERATURE	151	30	352	15	15
		TIME	300	87	113	16	10
		YIELD_OTHER	418	31	17	11	3
		YIELD_PERCENT	361	36	23	13	3
	WORKUP	TEMPERATURE	9	7	101	0	20
		TIME	0	0	43	0	13
		YIELD_OTHER	-	-	-	-	-
		YIELD_PERCENT	-	-	-	-	-
System			3162	763	2641	134	175

The results are consistent with the previous observations from the tables 8, 9 and 10. We can see REACTION_STEP classes performed better than the WORKUP classes. It is safe to say that, class imbalance plays a significant role in the miss-annotation of the instances. The results also show that the rule-based model significantly over annotates given the number of false positives. For example, the rule-based model (Run 2) identified 379 instances of the WORKUP-REACTION_PRODUCT event class with only four being true positives.

Table 14: Error analysis for the rule-based model where trigger words are trained with ChemPatent embeddings

Argument	Trigger	Entity	tp	fp	fn	fpm	fnm
ARG1	REACTION_STEP	OTHER_COMPOUND	40	1798	23	18	11
		REACTION_PRODUCT	351	75	101	10	3
		REAGENT_CATALYST	177	162	328	8	8
		SOLVENT	234	54	193	4	7
		STARTING_MATERIAL	217	128	494	15	9
	WORKUP	OTHER_COMPOUND	1501	171	249	54	73
		REACTION_PRODUCT	4	375	0	9	0
		REAGENT_CATALYST	0	40	0	9	0
		SOLVENT	2	25	0	5	0
		STARTING_MATERIAL	1	24	0	2	0
ARGM	REACTION_STEP	TEMPERATURE	450	131	53	29	15
		TIME	386	66	27	21	10
		YIELD_OTHER	350	74	85	11	3
		YIELD_PERCENT	326	55	58	11	3
	WORKUP	TEMPERATURE	89	45	21	13	20
		TIME	23	41	20	16	13
		YIELD_OTHER	0	367	0	10	0
		YIELD_PERCENT	0	325	0	8	0
System			4151	3957	1652	421	175

5 Conclusion

We trained three biLSTM+CRF models over different pre-trained word embeddings, as well as differently sized datasets. Results show that while these models did not outperform the baseline model when evaluating exact span matches, the models outperformed the baseline when evaluating in relaxed mode. A model trained using word embeddings trained over chemical patents performed best when evaluating in relaxed mode, while a model trained using biomedical word embeddings and a combination of the training and development datasets performed best when evaluated on exact span matches. Errors primarily occurred because of issues with the model distinguishing between different entity labels, such as models mislabeling entities annotated as OTHER_COMPOUND for more specific labels, like REACTION_PRODUCT or STARTING_MATERIAL. Additionally, the way that MedaCy predicts entity labels may have contributed to errors with labeling entity spans fully. Future work will focus on better distinguishing between different types of chemical compounds, as well as looking into models based on language models.

We used one CNN-based model and two rule-based models to extract events and according to the results, all three models outperformed the baseline model. Results show that the CNN-based method outperforms the rule-based methods, especially with the REACTION_STEP classes as those classes have more instances to train on. Meanwhile, as the rule-based methods do not require training instances to train they perform better with WORKUP classes. In the future, we plan to explore building a hybrid model with both CNN and rule-based methods to increase the performance.

References

1. Bort, W., Baskin, I.I., Sidorov, P., Marcou, G., Horvath, D., Madzhidov, T., Varnek, A., Gimadiev, T., Nugmanov, R., Mukanov, A.: Discovery of novel chemical reactions by deep generative recurrent neural network (2020)
2. Charles, P.: Project title. `https://github.com/charlespwd/project-title` (2013)
3. Gridach, M.: Character-level neural network for biomedical named entity recognition. Journal of biomedical informatics **70**, 85–91 (2017)
4. Huang, Z., Xu, W., Yu, K.: Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
5. Leaman, R., Gonzalez, G.: Banner: an executable survey of advances in biomedical named entity recognition. In: Biocomputing 2008, pp. 652–663. World Scientific (2008)
6. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
7. Nguyen, D.Q., Zhai, Z., Yoshikawa, H., Fang, B., Druckenbrodt, C., Thorne, C., Hoessel, R., Akhondi, S.A., Cohn, T., Baldwin, T., et al.: Chemu: Named entity recognition and event extraction of chemical reactions from patents. In: European Conference on Information Retrieval. pp. 572–579. Springer (2020)
8. Nguyen, T.H., Grishman, R.: Relation extraction: Perspective from convolutional neural networks. In: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing. pp. 39–48 (2015)
9. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d Alche-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019)
10. Pyysalo, S., Ginter, F., Moen, H., Salakoski, T., Ananiadou, S.: Distributional semantics resources for biomedical text processing. In: The 5th International Symposium on Languages in Biology and Medicine (2013)
11. Wang, K., Wang, L., Yuan, Q., Luo, S., Yao, J., Yuan, S., Zheng, C., Brandt, J.: Construction of a generic reaction knowledge base by reaction data mining. Journal of Molecular Graphics and Modelling **19**(5), 427–433 (2001)
12. Yoshikawa, H., Nguyen, D.Q., Zhai, Z., Druckenbrodt, C., Thorne, C., Akhondi, S.A., Baldwin, T., Verspoor, K.: Detecting chemical reactions in patents (2019)