Linguistically Motivated Question Classification

Alexandr Chernov **Volha Petukhova** **Dietrich Klakow**
Saarland University, Spoken Language Systems
Saarbrücken, Germany
{alexandr.chernov, v.petukhova, dietrich.klakow}@lsv.uni-saarland.de

Abstract

In this paper we describe a question interpretation module designed as a part of a Question Answering Dialogue System (QADS) which is used for an interactive quiz application. Question interpretation is achieved in applying a sequence of classification, information extraction, query formalization and query expansion tasks. The process of a question classification is performed based on a domain-specific taxonomy of semantic roles and relations. Our taxonomy was designed in accordance with the real spoken dialogue data. The SVM-based classifier is trained to predict the Expected Answer Type (EAT) with the precision of 82%. In order to retrieve a correct answer, focus word(-s) are extracted to augment the EAT identified by the system. Our hybrid algorithm for the extraction of focus words demonstrates the accuracy of 94.6%. EAT together with focus words are formalized in a query, which is further expanded with the synonyms from WordNet. The expanded query facilitates the search and retrieval of the information that is necessary to generate the system's responses.

1 Introduction

Any question answering (QA) system has to be able to give as precise as possible answers to natural language questions. In order to perform this task with a reasonably high accuracy, an adequate question interpretation is required. In the NLP field, this problem is often defined as the question classification. Due to the ambiguity of natural language utterances the task may become very complicated. For this reason the question classification phase has proven to be one of the most important parts of many QA system. If a question type is not correctly identified, the system will not be able to find the correct and/or complete answer. According to Razmara et al. (2007), correctly classified questions are answered correctly twice as often as misclassified ones.

The study conducted by Moldovan et al. (2000) set a new modern foundation in the QA task. An end-to-end open-domain QA system has been developed. In TREC-8[1], it achieved the highest result by demonstrating the accuracy of 77.7%. The designed system performs question processing, including question classification, focus and key words extraction, as well as the specification of an expected answer type.

In 2011, IBM Watson QA system (Ferrucci et al., 2010) won Jeopardy! quiz game, where it was able to beat two highest ranked players. The system includes a component responsible for the question analysis: the system needs to know what was asked in a question. Having this knowledge, the system generates candidate answers. In 2013 IBM made an attempt to adapt Watson QA to the healthcare domain (Ferrucci et al., 2013).

The scenario targeted in our application is comparable to the Jeopardy! quiz game. Our system, however, provides an interactive quiz game meaning that the returned answers are not just extracted information chunks or slot fillers, or database entries, but rather full-fledged dialogue utterances. The domain, on the other hand, is restricted to bibliographical facts about a famous person, and the player's task is to guess his/her identity by asking ten questions of various types. For such a close-domain, for the system to understand a question it is possible to narrow down the knowledge available to it. For example, structured knowledge bases can be used, e.g. Freebase[2]. They are however not complete to achieve sufficient cover-

[1] http://trec.nist.gov/pubs/trec8
[2] http://www.freebase.com/

age of factual information required for our game. Therefore, the content that the system operates on is a bigger collection of unstructured free texts, namely, Wikipedia articles[3]. This impacts search and retrieval tasks. As a consequence, the output of a question interpretation module should be a rather comprehensive query capturing various semantic information concerning events in question, entities involved in this event and their properties, and type of relations between entities and possibly between events. Thus, question interpretation is defined as a sequence of classification, information extraction, query formalization and query expansion tasks. Given the closeness of the domain, the system can operate on the basis of pre-defined domain-specific taxonomy of various semantic relations between different types of entities in order to compute an Expected Answer Type (EAT). The EAT is classified using statistical classifiers like Support Vector Machines (SVM) operating on multiple features, such as n-grams, part-of-speech and other syntactic information. The EAT is further augmented with question focus word(-s) information to determine the main event in question. Both, the EAT and focus word(-s), are formalized in a query which, on its turn, is expanded to cover as many as possible natural language variations.

The paper is structured as follows. Section 2 presents related work that has been reported in the area of general question answering and in question classification in particular. In Section 3 we outline performed experiments describing the data, tagset, features, algorithms and evaluation metrics that have been used. Section 4 reports on the experimental results, applying SVM on various feature combinations, to assess the automatic EAT classification. We also assess the semantic relations learnability by partitioning the training set and increasing a number of training instances in each next run. In Section 5 we describe an algorithm for automatic extraction of focus words. Section 6 explains how the query is generated and expanded. Section 7 summarizes our findings and outlines plans for the future work.

2 Related Work

Depending on the domain and task, QA systems may require different kinds of question type taxonomies. The main difference lies in the principle on which the question categorisation is performed.

[3]http://www.wikipedia.org

Lehnert (1986) developed a conceptual taxonomy with 13 conceptual classes (e.g. *causal antecedent, goal orientation, enablement, etc.*). This kind of categorization allows considering processes which occur within human memory during interpretation. Lehnert (1977) also pointed out that for the correct categorisation of ambiguous questions the context is very helpful.

Singhal et al. (1999) designed a very simple taxonomy based on the correspondence between question words and expected answer types. For instance, according to this taxonomy, questions containing *Who* or *Whom* belonged to the type *Person*. For more ambiguous words like *What* or *Which* the type of a question was identified by the head noun.

Li and Roth (2002) implemented a more advanced system. They created a hierarchical classifier relying on the answer type semantics, the taxonomy had 2 layers: 6 coarse classes (*abbreviation, entity, description, human, location, numeric value*) and 50 fine classes (subclasses of different coarse classes do not overlap). Using a hierarchical classifier they tried to get an increase in performance, but experimental results showed that the gained difference with a flat classifier turned out to be insignificant.

The system called Quarc performed a question categorisation relying exclusively on the presence of certain question words (e.g. *who, what, when, where, why*). For each question word the system had a set of heuristic rules which were applied to find out what kind of information an answer should contain. For example, *What*-questions may refer to objects (*What is on the picture?*), humans (*What was the name of the main character?*), or to time (*What year was America discovered in?*) (Riloff and Thelen, 2000).

Nowadays statistical machine learning is actively used for NLP tasks, also for the question classification. Many studies on machine learning indicate that there are no significant differences in performance of existing classification algorithms (Sebastiani, 2002). For example, Huang et al. (2008) applied classifiers based on linear SVM and Maximum Entropy models to the question classification problem. Almost identical accuracy has been achieved: 89.2% and 89.0% respectively. Panicker et al. (2012) used Naive Bayes and SVM classifiers for a comparable problem. Under different conditions the classifiers demonstrated similar results. However, the authors decided in favour of SVM because it was proved to be more effective for complex data.

Further, the question focus may be used to find an answer. Moldovan et al. (2000) defines the question focus as a word or a sequence of words which helps to identify what is asked in a question. Mikhailian et al. (2009) introduced two different types of the question focus:

1. Asking Point (AP) - the explicit question focus, e.g. in the question *Which books have you read?* the word *books* denotes AP;
2. Expected Answer Type (EAT) was used when the answer type was implicit but could be inferred from the information provided by the question, e.g. *person* is the EAT for the question *Who wrote "Pride and Prejudice"?*.

Focus words were applied as features to predict question types. Mikhailian et al. (2009) reported about accuracy of above 82%.

Ferret et al. (2001) defined the question focus as "a noun phrase that is likely to be present in the answer". The focus of a question consists of a head noun and a list of its modifiers. Their QALC system was able to correctly identify the focus for 85% of the questions from TREC10 dataset.

eatEntities	**29.21**		
creatorOf	9.74	partIn	2.46
activityOf	6.95	episodeOf	0.79
famousFor	3.16	interestOf	0.29
fieldOf	2.95	otherEntities	0.21
award	2.66		
eatHumanDescription	**28.76**		
title	11.9	nationality	1.46
name	7.49	religion	1.21
ageOf	2.08	gender	1.21
educationOf	1.66	otherHumanDescription	0.12
body	1.62		
eatHumanGroups	**10.61**		
memberOf	2.25	supporterOf	0.58
chargedFor	2.21	victimOf	0.25
employeeOf	1.79	causeOf	0.17
ownerOf	1.37	subordinateOf	0.12
founderOf	1.17	otherHumanGroups	0.04
superiorOf	0.62	chargeeOf	0.04
eatTime	**9.45**		
time	4.24	period	0.75
timeDeath	2.16	duration	0.67
timeBirth	1.62		
eatLocation	**9.4**		
loc	2.41	locActivityOf	0.92
locBirth	1.96	locDeath	0.83
locResidence	1.66	locFamousFor	0.29
locOrigin	1.33		
eatHumanRelations	**7.41**		
spouseOf	1.87	siblingOf	0.67
parentOf	0.96	friendOf	0.58
familyOf	0.92	enemyOf	0.54
childOf	0.83	otherHumanRelations	0.25
colleagueOf	0.79		
eatDescription	**4.12**		
typeOf	2.16	otherDescription	0.29
manner	0.75	definitionOf	0.21
reason	0.67	purpose	0.04
Multilabel	**1.04**		
fieldOf+spouseOf	0.12	activityOf+childOf	0.04
spouseOf+famousFor	0.12	spouseOf+gender	0.04
siblingOf+activityOf	0.12	spouseOf+founderOf	0.04
title+famousFor	0.08	spouseOf+award	0.04
title+childOf	0.08	otherHumanRelations+famousFor	0.04
title+spouseOf	0.08	title+loc	0.04
nationality+spouseOf	0.08	famousFor+otherHumanRelations	0.04
founderOf+activityOf	0.04		

Table 1: Distribution of semantic relation classes (in terms of relative frequencies in the corpus).

3 Experimental Set Up

3.1 Data and Tagset

It is generally known that spoken language differs from its written form in terms of grammaticality, syntax, vocabulary, etc. Our system is primarily focused on the spoken natural language processing. Unfortunately, publicly available corpora did not meet the requirements of our application.

In order to better understand the nature of spoken dialogue data and to obtain training data, the series of Wizard-of-Oz experiments were conducted. 338 dialogues were recorded, their total duration constitutes about 16 hours (Petukhova et al., 2014). 1342 unique questions were extracted and annotated with semantic relations. Two separate annotators were working on the labelling. To measure the agreement between them, we calculated Cohen's kappa score (Cohen, 1960) for all obtained labels. The kappa score equal to 0.85 was acquired, which indicated a very high degree of agreement between the annotators. Disputable questions were re-annotated together after a thorough discussion.

A preceding study on the question classification problem (see Faiz (2014) for more details) focused on automatically generated data. 1067 questions were obtained from the corresponding Wikipedia article using tool developed by Heilman (2011) and used for the training of an SVM-based classifier. The best precision of 80.18% was achieved on the combination of unigrams and bigrams of lemmas. We combined these two corpora, the resulting dataset contained 2403 (some questions were excluded due to the differences between the taxonomies).

We developed a hierarchical taxonomy of question types, which consists of two layers: coarse classes and fine classes. The full set of the defined relations is presented in Table 1 (see also Petukhova et al. (2014) for more details) with their relative frequency in the data (coarse classes are in bold).

3.2 Classifier

We used scikit-learn[4] (see Pedregosa et al. (2011) for more details), a machine learning library for

[4]http://scikit-learn.org

python, to build a question classifier based on the SVM algorithm and linear kernel function (linearSVC). Since we work in a quite specific domain, we could not obtain a separate dataset for testing. For this reason it was decided to apply a stratified 5-fold cross-validation. The number of folds was chosen based on the analysis of the data. According to Table 1, some classes in our dataset are under-represented. Dividing questions into 5 folds, we were able to equally distribute questions of each class (except for the ones represented by less than five instances).

Our classifier performs multi-class and multi-label classification. Thus, the classifier can handle questions containing several semantic relations which is often the case in real life situations.

We can use the hierarchical structure of our taxonomy to better discriminate between different question types. There are at least two possible ways of how it can be implemented:

1. Sequence of classifiers, where classifier#1 predicts coarse class labels and classifier#2 applies these labels as additional features.
2. Hierarchy of classifiers, where classifier#1 decides to which coarse class a question belongs and transfers it to the corresponding classifier trained specifically for these types of questions.

In our experiments we followed the first approach. According to (Li and Roth, 2002) who worked on a very similar problem there is no significant difference in performance between flat and hierarchical classifiers.

3.3 Features

No matter what learning algorithm or approach is applied, text-based features remain important for the classification task. The bag-of-words (BoW) approach, for example, is by far most widely used in text classification. This approach does not take into account the order of words and their co-occurrences. Therefore, apart from bow-models we constructed models based on bigrams, trigrams, and their combinations to assess their impact on the overall classifier performance. It is also of great interest to understand whether additional linguistic information helps to better discriminate between different classes.

The corpus of annotated questions was processed using the Stanford CoreNLP tools[5] to obtain part-of-speech and lemma information. In our experiments surface word forms, POS-tags, lemmas, as well as surface forms + POS-tags, lemmas + POS-tags, focus words and lemmas of focus words were used as features. Apart from that, we applied combinations of all the above mentioned features with coarse class labels to predict fine classes.

In order to extract focus words, we implemented an algorithm that preserves the main nominal phrase with the predicate, corresponding prepositions and conjunctions while removing everything else. The algorithm excludes stop words and stop phrases (from predefined lists), as well as some parts of speech (based on the Penn Tree Bank tagset[6] we remove existential *there*, interjections, interrogative pronouns and possessive endings), auxiliary verbs, and interrogative pronouns. Questions from the real dialogue data were manually annotated with focus words, which allowed to test this algorithm. It was able to extract focus words with the accuracy of 94.6%.

3.4 Evaluation Metrics

It is desirable, that in a quiz game the system provides the player with a correct answer, and rather acknowledge the fact if no answer is not found by generating utterances like "*Sorry, I do not have this information*".[7] In other words, to return the correct answer or acknowledge the fact that no answer is found is more important for the overall system performance than to return a wrong answer. That is why the precision for both question classification and answer detection tasks was more important than the recall. The precision metrics indicates how relevant the returned answers is to the question asked. Recall, by contrast, indicates how many relevant answers are returned by the classifier, which is not important information for the system to know, therefore disregarded in further evaluations. We calculated a weighted precision score taking into account the proportion of instances in each class. The weighted precision is computed by the following formula:

$$P_w = \frac{\sum_c P_c W_c}{\sum_c W_c}$$

[5]http://nlp.stanford.edu/downloads/corenlp.shtml

[6]http://www.cis.upenn.edu/∼treebank/

[7]To make the game more entertaining, the system can always play with strategies to turn a negative situation in a system's favour. For example, if no answer is found, the system may ask the player to put another question claiming that the previous one was not eligible for whatever reasons or the answer to it would lead to quick game end, or alike.

where P_c - precision for a certain class of questions, W_c - weight associated with that class (number of instances in a individual class).

3.5 Experimental Design

As a baseline it is common practice to use the majority class tag, but for our data sets such a baseline is not very useful because of the relatively low frequencies of the tags for many classes (see Table 1). Instead, we computed a baseline that is based on a single feature, namely, bag-of-words when training the Naive Bayes classifier. The baseline classifier achieved the precision of 56%. It was implemented using Multinomial Naive Bayes algorithm from scikit-learn (Pedregosa et al., 2011). Naive Bayes has been chosen for several reasons. Firstly, it is considered to be one of the basic classification algorithms. Secondly, it can be easily implemented. Thirdly, Naive Bayes is relatively simple and works quite fast.

In the first experiment we intended to establish how the classifier performs on the following features: surface word forms, POS-tags, lemmas, surface forms + POS-tags, lemmas + POS-tags, focus words and lemmas of focus words.

Second experiment is based on the assumption that the classifier should be able to predict coarse classes with a higher precision, since coarse classes are better represented in our data. We added coarse class labels as complementary features to the existing ones to predict fine classes. Labels were taken from the annotated data. Unfortunately, this is not a realistic setting, since the classifier can hardly predict coarse class labels with the precision of 100%.

In the third experiment, the classifier was trained on the actual predicted coarse-class labels instead of the annotated ones.

4 Experimental results

We have conducted three experiments, each time using different feature sets. Our classifier outperformed the baseline (X^2 (1, n = 2403) = 293.181, p<.05). The highest precision of 82% was achieved by the model which had been trained on unigrams+bigrams of lemmas. In most cases models based on unigrams+bigrams demonstrated significantly better results than unigram, bigram, or trigram models. It means that the word order is important for the classifier, but not very crucial. In Table 2 we summarize results from all of the experiments.

Features	**n-grams range**				
	1,1	**1,2**	**2,2**	**2,3**	**3,3**
Experiment 1					
Words	0.81	0.81	0.72	0.72	0.68
POS-tags	0.27	0.4	0.4	0.46	0.44
Lemmas	0.8	**0.82**	0.74	0.73	0.71
Words+POS-tags	0.8	0.81	0.71	0.71	0.68
Lemmas+POS-tags	0.8	0.82	0.73	0.73	0.71
Focus	0.75	0.74	0.63	0.59	0.31
FocusLem	0.76	0.76	0.65	0.61	0.39
Experiment 2					
Words+CA	0.86	0.86	0.8	0.79	0.71
POS-tags+CA	0.55	0.66	0.65	0.64	0.61
Lemmas+CA	0.86	0.87	0.81	0.81	0.76
Words+POS-tags+CA	0.85	0.85	0.8	0.79	0.7
Lemmas+POS-tags+CA	**0.87**	0.86	0.81	0.81	0.76
Focus+CA	0.82	0.82	0.72	0.68	0.5
FocusLem+CA	0.83	0.83	0.75	0.71	0.52
Experiment 3					
Words+CP	0.82	0.81	0.77	0.76	0.7
POS-tags+CP	0.41	0.6	0.59	0.6	0.56
Lemmas+CP	0.81	**0.82**	0.78	0.78	0.75
Words+POS-tags+CP	0.81	0.81	0.77	0.76	0.7
Lemmas+POS-tags+CP	0.81	0.82	0.78	0.78	0.75
Focus+CP	0.79	0.77	0.68	0.65	0.44
FocusLem+CP	0.79	0.79	0.71	0.68	0.48

Table 2: Precision of the classifier for fine classes (CA - coarse class labels from the annotated data, CP - coarse class labels predicted by the classifier).

In Experiment 1 models based on unigrams and unigrams+bigrams of surface word forms achieved the precision 81%, while models based on unigrams+bigrams of lemmas - 82%. These are the two best results in the Experiment 1. However, it is necessary to say that there is no significant difference in performance between these models (X^2 (1, n = 2403) = 0.5745, p>.05).

Deviations in performance between the unigrams and unigrams+bigrams of lemmas turned out to be statistically insignificant (X^2 (1, n = 2403) = 2.2640, p>.05). However, the model based on unigrams+bigrams is more precise than the one based on bigrams (X^2 (1, n = 2403) = 33.3749, p<.05).

Unfortunately, by adding POS-tags we did not get any improvements. Words+POS-tags and Lemmas+POS-tags feature sets accounted for the same maximal values: 81% and 82% respectively.

Using exclusively POS-tags as features, the classifier was able to achieve the precision of 46%. It is a very poor result in comparison to the unigrams+bigrams of lemmas (X^2 (1, n = 2403) = 522.6607, p<.05).

Surface word forms and lemmas of focus words demonstrate similar results (X^2 (1, n = 2403) = 0.4674, p>.05), achieving maximal precision of 75% and 76% respectively. These values are significantly lower than the results achieved by question surface word forms (X^2 (1, n = 2403) = 12.4608, p<.05) or by question lemmas (X^2 (1, n = 2403) = 18.1542, p<.05).

We can see that in Experiment 2 unigrams, unigrams+bigrams of Words+CA and unigrams+bigrams of Lemmas+CA perform almost equally well (X^2 (1, n = 2403) = 0.7440, p>.05), demonstrating the highest precision of 86% and 87% respectively. There is no significant difference between unigrams and unigrams+bigrams of lemmas (X^2 (1, n = 2403) = 0.7440, p>.05). The difference is significant for unigrams+bigrams and bigrams of lemmas (X^2 (1, n = 2403) = 24.1152, p>.05), and for unigrams+bigrams and bigrams of surface words forms.

Combinations with POS-tags again were not beneficial for the classification process. They did not show any improvements.

Comparing the results of Experiment 2 with the results of Experiment 1, we may conclude that by adding coarse class labels as additional features a significantly higher precision was achieved.

In Experiment 3 we used coarse class labels predicted by the classifier (on unigrams+bigrams of lemmas) and did not observe any significant difference in comparison to the results from Experiment 1. Unigrams+bigrams of lemmas, unigrams+bigrams of Words+POS-tags and Lemmas+POS-tags demonstrated absolutely identical results.

As for separate classes, questions of the most prevailing classes were identified with a very high precision: *title* - 85%, *creatorOf* - 81%, *name* - 89%.

The most frequent questions in our corpus are related to the professional activity of a person. Most of the time this kind of questions belong to the class *acitivityOf* or to the class *title*. They turned out to be very similar: for example, by asking *What do you do for a living?*, the player expects to get as a potential answer either the description of a particular professional activity or the name of a title (position) the person holds. Moreover, very often the player does not care which of them will be chosen, both answers will be correct. As consequence, the classifier can confuse the classes *acitivityOf* and *title* with each other.

As we expected, the classifier achieved the best results by using lexical clues, i.e. the presence of absence of certain words is a strong feature to determine to which class or classes a question will be assigned. Unfortunately when a question contains words shared by questions belonging to different classes, it may cause prediction errors. For example, the classifier may assign several labels instead of one and vice versa. Based on the analysis of misclassified instances, we can tell that a question will receive more than one label, if wording representative for two (or more) classes is observed and extracted as features.

The analysis of false predictions indicates that most of them were caused by the imbalanced training set. There are also no strict borders between some classes. Questions with multiple labels are under-represented. According to Table 1 they comprise only 1.04%.

By applying coarse class labels as additional features we tried to get a higher precision. Unfortunately, it worked only when these labels were taken from the annotated corpus. The classifier was able to predict coarse class labels with the average precision of 90% (see Table 3). However, it was not enough to make the actual predicted labels useful for the next classifier.

Classes	**Precision**
eatEntities	0.86
eatTime	0.97
eatHumanRelations	0.92
eatHumanDescription	0.9
eatLocation	0.91
eatDescription	0.86
eatHumanGroups	0.88
avg/total	0.9

Table 3: Precision of the classifier for coarse classes (unigrams+bigrams of lemmas).

The precision for all coarse classes is already relatively high. Questions of the class *eatTime*, for example, were correctly identified in 97% of cases. It may be very problematic to make further improvements.

To explore learnability of the best performing classification model and to evaluate how the size of the training set affects the classifier's results, we divided the corpus of annotated questions into 20 parts. All partitions, except for the first one, contained the equal number of questions. Different question types were equally distributed among the partitions. We started with the training set consisting of 636 questions and gradually increased its size. Based the obtained results, the learning curve has been plotted presented in Figure 1.

As we can observe, the precision rose almost steadily until the size of the training set became

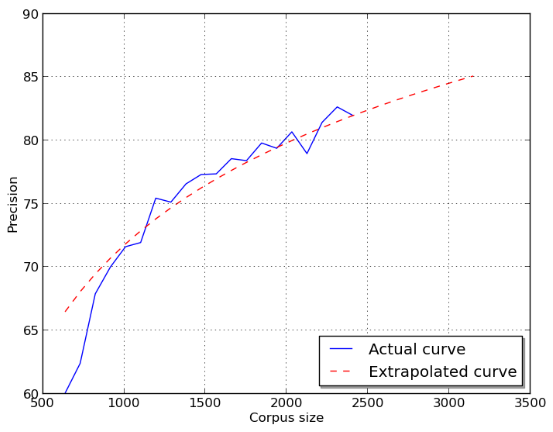

Figure 1: Learning curve.

bigger than 2000 questions. The growth stopped at the precision of around 81%. It was followed by a decrease of 2%. After that the precision grew by 3%, and then again dropped about 1%. These fluctuations, as we believe, were caused by the quality of the data.

The growth slows down gradually, i.e. in the range from about 700 till 1200 questions the precision increased from 65% till 75%, while to get another 5%, the classifier required 800 additional questions. Taking these calculations into account, we were able to obtain the formula, which allowed to extrapolate the learning curve:

$$y = 1.164396665 * 10^{-1} * ln(x) - 8.691998379 * 10^{-2}$$

We came to the conclusion that the classifier will need the training set including approximately 3100-3200 questions to achieve the precision of 85%. Thus, getting more data may potentially improve the performance of the classifier. However, given the obtained learnability results and since data collection and its annotation is a very time consuming task, the efforts may be better spent to explore other approaches additional to machine learning, e.g. pattern matching and bootstrapping from collected examples.

5 Question Focus Extraction

In line with Moldovan et al. (2000), the question focus describing the main event is typically expressed by a verb or eventive noun. Despite the fact that the focus is semantically defined, we use the knowledge of syntactic structures, since syntactic parsers are mature enough comparing to semantic ones to be used reliably. The following procedures have been applied to automatically extract focus words:

1. **Auxiliary verbs elimination**. OpenNLP chunker[8] detects VP-chunks. There is a predefined set of rules helping to identify which of them contains an auxiliary:
   - *Combinations of adjacent chunks like VP+NP+VP are glued together. The first verb in such combinations is usually an auxiliary, checked in the list of auxiliaries.*
   - *If there is only one verb in a sentence than it is not an auxiliary verb.*
   - *If a long chunk contains several verbs, at least one of them should be an auxiliary, checked in the list of auxiliaries.*
2. **Removal of opening and closing phrases using regular expressions**. For example, *"Could you tell me what are you doing for living?"*, *"You are an American, aren't you?"*.
3. **Stop words and stop phrases removal**.
4. **Postprocessing**. Removal of extra spaces, conjunctions left at the beginning or at the end of the focus.

This algorithm demonstrated the accuracy of 94.6% when evaluating on the manually annotated reference data.

6 Query Generation and Expansion

Question	What do you do as a job?
Focus words	do as job
Expanded focus	do [make, perform, cause, practice, act], as, job [activity, occupation, career, employment, position]
EAT	Title_do(do as job)
Query	(Z, E, ?X) :: Title_do(Z, doAs, ?job) :: QUALITY(String) :: QUANTITY(List) :: FOCUS(do as job)
Expanded query	(Z, E, ?X) :: Title_do(Z, doAs, ?job) :: QUALITY(String) :: QUANTITY(List) :: FOCUS(do [make, perform, practice, act], as, job [activity, occupation, career, employment, position])

Table 4: Example of an expanded query.

Query generation is the last data processing operation that is performed in the question interpretation module. The query is generated according to the pre-defined set of rules. It captures the results of the question classification (labels) process as well as the extracted focus words and transfers this information to the next module.

[8]https://opennlp.apache.org

The query generation processes, the semantic representation of its components in particularly, partially based on the Discourse Representation Theory (DRT) (Kamp and Reyle, 1993). It incorporates semantic information that is necessary to find the correct answer. Table 4 demonstrates an example of such a query.

In natural languages the same message has a number of realizations. So far, our QA system misses many answers when the answer is expressed by different lexical units. To solve this problem, we used WordNet[9] synonyms to elaborate the extracted question focus words.

7 Conclusions and Future Work

To implement a question classifier for our system, we used SVM algorithm. This algorithm performs quite accurate classification, has a mechanism to avoid overfitting, can be customized by changing its kernel function, and is able to handle high dimensional spaces.

We have annotated a corpus of questions, which will become publicly available in the nearest future. Although our corpus has been designed for a quite specific gaming application, it may be of interest for researchers working on various topics. Regardless of the domain, annotated spoken dialogue data may help in studying of different linguistic phenomena such as, for example, ellipsis or co-reference resolution.

The corpus has been used as a training set for a question classifier. The classifier was able to predict EAT with the precision of 82%. This result was achieved by the model based on the unigrams and bigrams of lemmas.

Having analysed misclassified questions, we drew several conclusions. First, the classifier confuses semantically similar classes. Second, it has difficulty to identify EATs for under-represented classes. Third, questions simultaneously belonging to several classes were often misclassified.

Additional to the EAT, focus words are important to find correct answers. Moldovan et al. (2000) defines the question focus as a word or a sequence of words which helps to identify what is asked in a question. In order to automatically extract focus words, we have implemented an algorithm that performs with the accuracy of 94.6%.

Once EAT and focus words are specified, this information needs to be formalized in a form of a query, in order to be processed by next modules, in particular for answer retrieval and generation, and for dialogue manager to update the latest information state and decide on further dialogue actions. To address this problem, the question classification module generates a query which incorporates various linguistic information such as one or multiple semantic relations, events, named entities mentioned in a question, the entity or event for which information (slot filler) has to be found.

Our findings confirmed that by increasing the training set we can slightly improve the precision of the classifier. However, due to the specificity of our data, this task becomes quite difficult. Wizard-of-Oz experiments involve human participants and are conducted in a controlled setting. All participants have to be instructed in advance. After experiments dialogue data should be analysed, transcribed, and manually annotated by at least several trained annotators. The listed actions require considerable amount of efforts and time.

The easiest way to achieve a higher precision is probably to increase the number of instances for the under-represented classes. Of course, it is impossible to force the users to ask only certain types of questions. However, new instances can be generated based on the existing ones using bootstrapping. The training set, which has been used to learn the classifier, is unbalanced. Ideally, all question types should be equally represented.

It is also possible to apply bootstrapping to generate synonymous questions for the whole corpus. In this case we will not discover any new phenomena, but we will get a better lexical coverage.

By querying search engines we can extract questions that match regular expressions. However, it should be noted that not all questions types can be encoded using regular expression. Data obtained in such a way may require some manual post-processing.

While testing/evaluating with the system, players produce a lot of questions. Saving each gaming session could help to enrich the training set.

The analysis of false predictions suggests that the taxonomy requires some refinements. Many classes were never used during the annotation. Certain classes appeared to be very similar to other classes or simply too general. They should be either merged together of divided into several more specific subclasses respectively.

[9]http://wordnet.princeton.edu

References

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

H. Faiz. 2014. Question classification module for question answering gaming application. Master's thesis, Saarland University, Germany.

O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, L. Monceaux, R. Robba, and A. Vilnat. 2001. Finding an answer based on the recognition of the question focus. In *TREC*.

D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, et al. 2010. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79.

D. Ferrucci, A. Levas, S. Bagchi, D. Gondek, and E. T. Mueller. 2013. Watson: Beyond jeopardy! *Artif. Intell.*, 199:93–105.

M. Heilman. 2011. *Automatic Factual Question Generation from Text. PhD thesis*. Carnegie Mellon University, USA.

Z. Huang, M. Thint, and Z. Qin. 2008. Question classification using head words and their hypernyms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 927–936, Stroudsburg, PA, USA. Association for Computational Linguistics.

H. Kamp and U. Reyle. 1993. From discourse to logic. Introduction to modeltheoretic semantics of natural language, formal logic and Discourse Representation Theory. *Studies in Linguistics and Philosophy*, 42.

W. Lehnert. 1977. *The Process of Question Answering. Research Report No. 88 [microform] / Wendy Lehnert*. Distributed by ERIC Clearinghouse [Washington, D.C.].

W. Lehnert. 1986. A conceptual theory of question answering. In Barbara J. Grosz, Karen Sparck-Jones, and Bonnie Lynn Webber, editors, *Readings in Natural Language Processing*, pages 651–657. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

X. Li and D. Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

A. Mikhailian, T. Dalmas, and R. Pinchuk. 2009. Learning foci for question answering over topic maps. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 325–328, Stroudsburg, PA, USA. Association for Computational Linguistics.

D. Moldovan, S. Harabagiu, A. Harabagiu, M. Pasca, R. Mihalcea, R. Girju, R. Goodrum, V. Rus, and I. Background. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL-2000)*, pages 563–570.

A. D. Panicker, A. U, and S. Venkitakrishnan. 2012. Article: Question classification using machine learning approaches. *International Journal of Computer Applications*, 48(13):1–4, June. Published by Foundation of Computer Science, New York, USA.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

V. Petukhova, M. Gropp, D. Klakow, G. Eigner, M. Topf, S. Srb, P. Moticek, B. Potard, J. Dines, O. Deroo, R. Egeler, U. Meinz, and S. Liersch. 2014. The DBOX corpus collection of spoken human-human and human-machine dialogues. In *Proceedings of the 9th Language Resources and Evaluation Conference (LREC)*. ELDA, Reykjavik, Iceland.

M. Razmara, A. Fee, and L. Kosseim. 2007. Concordia university at the TREC 2007 QA Track. In *TREC*.

E. Riloff and M. Thelen. 2000. A rule-based question answering system for reading comprehension tests. In *Proceedings of the 2000 ANLP/NAACL Workshop on Reading Comprehension Tests As Evaluation for Computer-based Language Understanding Sytems - Volume 6*, ANLP/NAACL-ReadingComp 00, pages 13–19, Stroudsburg, PA, USA. Association for Computational Linguistics.

F. Sebastiani. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, March.

A. Singhal, S. P. Abney, M. Bacchiani, M. Collins, D. Hindle, and F. C. N. Pereira. 1999. AT&T at TREC-8. In *TREC*.