Question Analysis for a Closed Domain Question Answering System

Caner Derici[1], Kerem Çelik[1], Ekrem Kutbay[2], Yiğit Aydın[2], Tunga Güngör[1], Arzucan Özgür[1], and Günizi Kartal[2]

[1] Boğaziçi University
Computer Engineering Department
[2] Educational Technology
Bebek, Istanbul, 34342
{caner.derici,kerem.celik2,ekrem.kutbay,yigit.aydin, gungort,arzucan.ozgur,gunizi.kartal}@boun.edu.tr

Abstract. This study describes and evaluates the techniques we developed for the question analysis module of a closed domain Question Answering (QA) system that is intended for high-school students to support their education. Question analysis, which involves analyzing the questions to extract the necessary information for determining what is being asked and how to approach answering it, is one of the most crucial vcomponents of a QA system. Therefore, we propose novel methods for two major problems in question analysis, namely focus extraction and question classification, based on integrating a rule-based and a Hidden Markov Model (HMM) based sequence classification approach, both of which make use of the dependency relations among the words in the question. Comparisons of these solutions with baseline models are also provided. This study also offers a manually collected and annotated vgold standard data set for further research in this area.

1 Introduction

Question Answering (QA) systems aim to produce automatically generated answers for questions stated in natural languages. The drastic improvements in the Natural Language Processing (NLP) and Information Retrieval (IR) techniques in the past decade have led to the development of prominent QA systems, some of which are available for public use, such as AnswerMachine[1] and WolframAlpha[2]. It has even been possible to develop a QA system that can compete on a TV show against human opponents [8].

Building a fully capable QA system, however, has difficulties mostly due to numerous challanging sub-problems that need to be solved, such as question analysis (involving pre-processing and classification of questions), information retrieval, cross linguality and answer generation (involving answer extraction and formulation), along with some lower level subtasks, such as paraphrasing, common sense implication or reference resolution. In addition, the architecture of a QA system, as well as the techniques employed usually

[1] http://theanswermachine.tripod.com/
[2] http://www.wolframalpha.com/

A. Gelbukh (Ed.): CICLing 2015, Part II, LNCS 9042, pp. 468–482, 2015.
DOI: 10.1007/978-3-319-18117-2_35

depend on factors such as question domain and language. Many researchers have tackled the individual problems involved in such systems separately. While some are considered to be solved, the majority of the problems are still open to further research [9,1].

This study attempts to solve the first problem of a QA system, question analysis. The overall system is developed for Turkish-speaking high-school students to enable them to query in their natural language any question chosen from their course of study. Note that there is virtually no upper bound in the number of possible query frequency, as the system is intended for use by virtually all high schools in Turkey. Therefore in order for the system to be practically usable, besides accuracy, the overall architecture should be carefully designed, where each module is comprehensively analysed and evaluated individually. In this study, we present the development and evaluation of the first module, namely, question analysis in the pipeline of our system, intended for use in the prototype domain of Geography. The primary concern in question analysis is to extract useful information from a given question to be used in subsequent modules to finally generate a correct response. In particular, the information that indicates a certain type or a central property of the entity being asked, along with a classification of the question into pre-determined classes from the domain helps to reduce significantly the size of the work space of the further stages in the system such as information retrieval or candidate answer generation.

In the following example, the information indicating that the name of a plain is asked, which we refer to as the *focus*, and the classification *ENTITY.PLAIN* helps us to navigate around these concepts in the knowledge base, searching the answer.

"Türkiye'nin en büyük [*ovasının adı*] *nedir?"*

"What is the [*name of the*] *largest* [*plain*] *in Turkey?"*

For focus extraction, we developed a rule-based model, along with a Hidden Markov Model (HMM) based statistical model. We investigate the accuracy of the combination of these two in focus extraction. Additionally, for question classification, we show that a rule-based model is more successful in finding coarse classes than a tf-idf based bag-of-words baseline model that utilizes the frequencies of the words in a question.

Developing such a question analysis module, let alone a QA system for Turkish is especially challenging because it is an agglutinative language with a morphologically rich and derivational structure. For this reason, we pre-process the questions by performing morphological analysis and disambiguation, as well as dependency parsing using the Turkish NLP Pipeline [16,6,15]. Morphological analysis and disambiguation produces the root forms of the words and their part-of-speech (POS) tags. Dependency parsing produces the dependency relations among the words in a given sentence. The tags that are used by the dependency parser are defined in the Turkish Dependency TreeBank, which includes tags such as *SUBJECT*, *OBJECT*, *SENTENCE*, *MODIFIER*, *CLASSIFIER*, *POSSESOR*, and etc [6,7].

We propose a novel approach for question classification and focus detection, based on integrating a rule-based method with an HMM-based sequence classification method, for a closed-domain QA system. Additionally, we contribute the first manually collected and annotated gold standard question analysis data set for Turkish. The implementation

codes and the gold standard Turkish question data will be publicly available for reproducability and further research[3].

2 Related Work

A fundamental task in a QA system is determining the type of the answer, and its properties and possible constraints. Given a query stated in a natural language, a QA system often extracts some immediate information such as the question class (e.g. *what, who, when, etc.*) based on the pre-determined answer types [4]. Recent state-of-the-art techniques for question classification often involve statistical approaches [12,13]. Additionally, some QA systems are in general more semantics oriented, and construct a knowledge base directly from raw question texts [10]. However, these systems determine only the type of a given question. They do not further determine, for example what type of entity is being asked, which would narrow down the search space significantly.

One approach in simulating a question analysis is to use general purpose search engines. One of the earliest studies that employs such a strategy, is an open-domain QA system, AnswerBus [19]. AnswerBus employs a bag-of-words strategy, where search engines are scored based on the number of hits they returned for each word. The total score of a search engine for a particular question is the sum of the hits returned for each word in the question. Based on their total scores, the best search engine is determined as the most appropriate knowledge source for answering the question. However, AnswerBus does not use any semantic information, nor does it extract any information to build a more competent answering strategy.

The first successful Turkish factoid QA system used a hybrid approach (both rule-based and statistical), not for question analysis, but for providing a direct answer by matching surface level question and answer patterns [5]. It doesn't employ any explicit question analysis, other than extracting the predefined question and answer patterns.

Inspired by its significant success, our system adapts its strategies for question analysis among the ones that are employed in one of the most powerful QA systems, IBM's Watson [11]. For analysing a given question (i.e. clue), Watson extracts firstly a part of the clue that is a reference to the answer (*focus*); second, it extracts the terms that denote the type of the entity asked *(lexical answer type, LAT)*; third, the class of the clue (*QClass*); and finally some additional elements of the clue (*QSection*) should it need special handling. Lally et al. [11] extensively evaluate the significance of distilling such information to produce correct answers. To extract these information, Watson mostly uses regular expression based rules combined with statistical classifiers to assess the learned reliability of the rules. Note that, the sole purpose of Watson is to win the Jeopardy! game, a well-known television quiz show where the quiz questions are presented as free formatted "clues", rather than complete question statements, rendering Watson's analysis methods specific to the Jeopardy! game. In a closed-domain QA system, on the other hand, it is sufficient to extract only *LAT* and *QClass* in order to analyse a complete question, since in a complete question sentence, what Watson refers to as the *focus* is often the question word (e.g. "What" in the example in Section 1). Therefore, the real

[3] `https://github.com/cderici/hazircevap`

focus of a question, what we refer to as the *focus* is closest in reality to what Watson refers to as *LAT*. In this regard, our definition of the focus is:

"the terms in the question that indicate what type of entity is being asked for".

A study most relevant for our question analysis is conducted by [3], where rule-based and statistical methods are utilized together to extract the question focus in an open-domain QA system. In this study, a binary classification using Support Vector Machines (SVM) is performed on words in the English questions that are parsed using a constituency parser. Further, experts with manually tailored rules are used to identify the different features which are then deployed in the SVM. In contrast, our analysis separately uses both rule-based and statistical models to extract the focus. It also performs question classification for Turkish questions that are parsed using a dependency parser. Additionally, a sequence classification is performed using a Hidden Markov Model (HMM) based algorithm, whose results are then combined with the results of the rule-based experts to produce the final focus. Unfortunately, our study is incompatible for comparison with this study. Firstly, because the definition of the focus in [3] depends on a constituency parser and a coreference resolver, which currently do not exist for Turkish. Therefore, it is neither possible to define equivalent rules for the English dataset, nor to apply the techniques proposed in [3] to the Turkish dataset.

3 System Architecture

Although the main technical contribution of this study is the methodology (i.e. the combination of the rule-based and statistical models), one of the tenets of this paper is to be an introduction of the QA system, upon which this analysis module resides and to be a starting point for the development of the subsequent modules. Consequently, this section introduces the general architecture of the system, as well as the way in which the question analysis module connects to it.

The overall architecture of the system is designed in concordance with the DeepQA technology, introduced in [8]. The primary principle in DeepQA is to have parallel units with multiple sub-modules that produce different candidate results for each sub-problem, which are then scored according to the evidence collected by trained machine learning models. Then the most likely candidate is returned as the final answer.

After question analysis, the extracted *focus* is used in the *Information Retrieval* module to fetch the relevant knowledge units[4] that are pruned and refined by the *QClass*. These relevant units are then fed to the *Candidate Answer Generation* module that has multiple different information retrieval algorithms to produce all possible relevant answer units. For each candidate answer unit, syntactic and semantic evidence units are collected, which are then used to score the candidate answers, the ones having low scores are pruned. Finally, the strong candidates are synthesized into the final answer set, where the most likely answer is fed to the answer generation module along with the other top k answers for providing optionality.

[4] We refrain from referring to these units as "documents", as we do not limit the format in which the knowledge is represented.

3.1 Question Analysis Module

The Question Analysis module consists of three parallel sub-modules, the *Distiller*, *HMM-Glasses* and *ClassRules*, illustrated in Figure 1. The first two modules are for extracting the question's *focus*, whereas the third module is for determining the most likely classification of the question *(QClass)* into the pre-defined question classes.

The *focus* indicates what exactly the given question is asking, and what type of entity it is. In the example in Section 1, the focus is the collection of these parts[5] of the question: "ovasının adı" *(name of a specific plain)*, since the question asks for a name. In particular, it asks the name of a plain. Therefore, the phrase "ova adı" *(name of a plain)* can be constructed even syntactically from the phrase "ovasının adı" *(name of a specific plain)*, since we already have the morphological roots attached to the question parts. Because "ova"*(plain)* is the root, and "sı" and "nın" are possessive suffixes which together mean: *"a name of a plain of"*. The *QClass* for this question is *ENTITY* (see Table 2).

In the following example, the *focus* is the parts "denizci kimdir" *(Who is the sailor)*, and the *QClass* is *HUMAN.INDIVIDUAL*. The rationale for the *focus* is that the question asks for a person's name, and it is known that the person is a sailor. Observe that we omit the distinctive properties of the entity in the question (e.g. the first sailor), because at this point, we are mostly interested in "is a" and "part of" relations that indicate a certain type of the entity. The remaining properties are used by the subsequent modules of the system to semantically prune both the *relevant knowledge units* and the *candidate answers*.

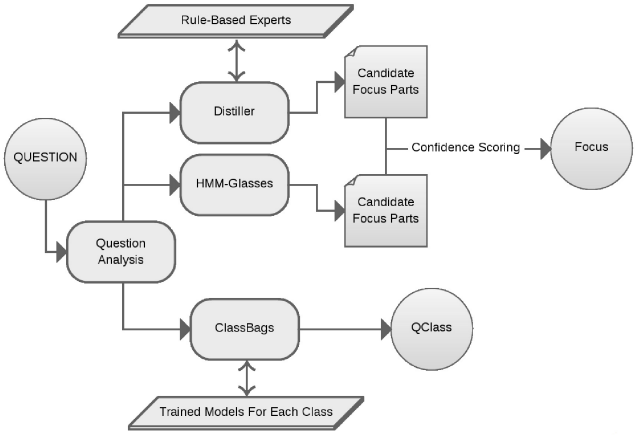


Fig. 1. Queestion Analysis Module

[5] Note that, we refer to each word of the question as a "part". A part represents a single word in the question that has been annotated with extra information such as its morphological root, part-of-speech, dependency tag, and etc.

"Dünyayı dolaşan ilk [*denizci kimdir*] *?"*

" [*Who is*] *the* [*sailor*] *that first circumnavigated the Earth?"*

4 Methodology

For *focus* extraction, we have a fastidious rule based focus extractor, the *Distiller*, with specifically tailored rules over the dependency trees for almost all types of factual questions in the Geography domain, and an HMM based classifier, *HMM-Glasses*, which uses a variation of the Viterbi algorithm [17] that essentially renders it somewhat more liberal than the Distiller to a certain extent. Other than one common trait, that is operating on the dependency relations among the words in the question, their approaches to the main problem (i.e. to extract the focus) are based on completely different principles in different levels of resolution. This distinction is critical to our methodology, since it provides the necessary insight for the model to efficiently handle languages with rich derivational structure, such as Turkish. At this point, a delicate balance is required for the combination of these models. For this purpose, we take into account the individual confidences of both the Distiller and HMM-Glasses, rendered through their individual performances over the training dataset. Additionally, for the classification of the question into predetermined classes from a certain domain (Geography in our case), we have a rule-based classifier, which extracts the coarse class by manually constructed phrase-based rules.

4.1 Focus Extraction

Distiller. We observed that in our selected domain of Geography, there are certain patterns of question statements (based on the predicate), common to the majority of the questions. We identified each such pattern (*question type*) and defined manually sets of rules (*experts*) for the extraction of the focus from the dependency parse tree of each question. We call this sets of rules together, *The Distiller*.

Currently we have seven rule-based experts, along with a generic expert that handles less frequent cases by using a single generic rule. The primary reason of the inclusion of a generic expert is data scarcity. However, we prefer to make it optional, because having a specific general expert along with a finite number of experts may result in a penalized precision as opposed to more or less increased recall, depending on the data set size, which may not always be a desirable option in practice. All experts and their question frequencies in the data set are given in Table 1.

The rules contain instructions to navigate through the dependency tree of a given question. For example, the rule for the "nedir"*(what is . . .)* expert, and the rule for the "verilir"*(. . . is given . . .)* expert, as well as the generic rule are as follows (examples provided in Figure 2).

nedir:*(what is . . .)*

- Grab the *SENTENCE* in the question
- Grab and traceback from the *SUBJECT*, and collect only *POSSESSOR* and *CLASSIFIER*

verilir: *(. . . is given . . .)*

- Grab the *SUBJECT* of the *SENTENCE* in the question
- Grab and traceback from the first degree
DATIVE.ADJUNCT of the *SENTENCE*, and collect only the firstdegree *MODIFIER*

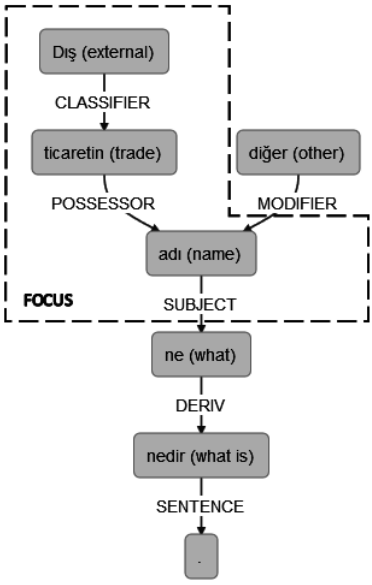
generic:

- Grab the *SUBJECT* of the *SENTENCE* in the question
- Traceback from the *SUBJECT*, and collect the first degree *POSSESSOR* and/or *CLASSIFIER*, along only with their *POSSESSOR* and/or *CLASSIFIER*

Every rule-based expert has a confidence score based on its performance for extracting the correct focus parts from the questions belonging to its expertise. This score is used to indicate the reliability of the expert's judgement later when combining its result with the *HMM-Glasses*. The confidence scores, along with the focus parts of a question $\mathbb{Q}$ are reported by both the *Distiller* and the *HMM-Glasses* in the format of triplets:

$$\langle fpt, fpd, fpc \rangle_n$$

where $n \in \{1..|\mathbb{Q}|\}$ [6], fpt stands for *focus part text*, fpd is *focus part dependency tag* and fpc denotes *focus part confidence* score. Both models produce such triplets for



Dış ticaretin diğer adı nedir?
What is the other name of external trade?

Fig. 2. *nedir* expert tells that the focus of this question is "a name of external trade"

[6] $|\mathbb{Q}|$ denotes the number of words in the question $\mathbb{Q}$.

Table 1. Experts and their question frequencies in the training data

Expert Type	*Frequency (%)*
generic	25.6
hangi *(... which ...)*	19.5
nedir *(what is ...)*	15.0
denir *(... referred to as ...)*	9.6
kaç *(how many ...)*	9.6
verilir *(... is given ...)*	7.2
hangisidir *(which one is it ...)*	7.2
kadardır *(how much ...)*	6.3

each focus part that they extracted. However, there is a significant distinction in the way that the confidences are reported for each part of the extracted focus between the rule-based and the statistical models. As explained in detail in Section 4.1, *HMM-Glasses* work on individual parts of the question, while the *Distiller* extracts sub-trees from the dependency tree of the question. Therefore, the *Distiller*'s resolution is not big enough to consider the individual probabilities for each part to be in the focus. Thus the *Distiller* produces a collection of parts as the *focus*, along with a single confidence score (*total confidence score*) reported by the expert in charge, this is mapped to fpc scores of all parts, rendering all parts in the focus equal from the *Distiller*'s perspective.

HMM-Glasses. *HMM-Glasses* models the focus extraction as a HMM and performs a sequencial classification on the words in the question using the Viterbi algorithm. Having only two hidden states, namely *FOC* (i.e. the observed part is a focus part) and *NON* (i.e. the observed part is not a focus part), it treats each question part as an observation, and decides whether the observed part is a part of the focus of the question.

We first serialize the dependency tree of the question and feed the algorithm the serialized tree. Serialization (or encoding) of a tree is to systematically produce a sequential representation of it, which is mostly employed in the fields of applied mathematics, databases and networks [18,14]. Evidently the method with which the tree is serialized has an observable influence on the characteristics of the algorithm's results. We investigated this effect with two general serialization approaches, and empirically tested it (see Section 6). Common approaches in tree serialization try to efficiently serialize the tree within the information theoretical resource bounds (in terms of time and space), while taking into account also the deserialization process [2]. On the other hand, we are only concerned with the coherency of the tree structure. In other words, the dependency relations should be consistent among all the serialization methods. Therefore, we considered the simplest possible methods, *forward mode* and *backward mode*.

Forward and Backward Modes. While constructing the sequence from the dependency tree in forward mode, left children (according to the reverse visualization of the dependency tree) take precedence over the right children to be taken into the sequence.

Therefore, the left-most branch is taken first, then the branch on its immediate right is taken, and so on. Finally the parent is added. Backward mode is simply the other way around, where the right children take precedence over the left children. Any difference in serialization changes the whole learning process, thereby renders the learned features unique to a particular serialization. This therefore provides a noticeable diversity in the characteristics of the learning, depending on the serialization method. Below are the serializations of the question in Figure 2. Recall that we only consider the morphemes of the words (i.e. stripped from all the adjuncts).

forward serialization (->)

Dış	ticaret	diğer	ad	ne
(external)	*(trade)*	*(other)*	*(name)*	*(what)*
FOC	FOC	NON	FOC	NON

backward serialization (<-)

ne	ad	diğer	ticaret	Dış
(what)	*(name)*	*(other)*	*(trade)*	*(external)*
NON	FOC	NON	FOC	FOC

Essentially, forward mode serialization corresponds to reading the question from left to right (or start to end), while backward mode corresponds to reading it from end to start. Different serialization approaches potentially allow ensembles of various kinds of models, handling different parts of the question as they have learned different features of the data while training. Therefore, a more complex model can be obtained by combining multiple *HMM-Glasses* having different serialization approaches.

We model the focus extraction problem as a HMM by firstly computing the prior probabilities of our hidden states (i.e. *FOC* and *NON*), and secondly learning the probabilities from the given set of serialized questions as follows

$$a_{jk} = P(t^j|t^k) \qquad b_{ij} = P(w_i|t^j)$$

where a_{jk} represents the probability of being in state t^j given the previous state is t^k, and b_{ij} indicates the probability that the current observation is the word w_i given that the current state is t^j. Decoding is performed using the Viterbi algorithm, where the states correspond to the nodes in the produced Viterbi path indicating the most likely judgements for each part to be a focus part of the question. Further, the observation probabilities b_{ij} are used as confidence scores (i.e. fpc) in the triplets. Recall that all results are reported as triplets (see Section 4.1).

Dependency Tags vs. Word Texts. In all parts of the question analysis, taking advantage of the dependency relations among the words in the question whenever possible has prominent benefits, compared to mere syntactic approaches for languages with a rich derivational structure, where for instance possible long distance relationships in

the question statement can easily be determined. Therefore, the very first design of the *HMM-Glasses* was planned to learn and evaluate the dependency tag sequence of a question, which essentially corresponds to learning the tree shape, rather than the sequence of words. However, this approach mislead the model, as there are some tags that occur more frequently in questions than others, as for example a question often has only one *SENTENCE* tag, while it has lots of *MODIFIER* tags. More importantly, the focus is often a small part of the question. Thus, for example, the judgement of whether a *MODIFIER* part is a focus part is strongly biased by the fact that the number of cases a *MODIFIER* is a NON will be orders of magnitude higher than otherwise. Furthermore, working with the normalized frequencies requires a lot of training data for the model to have a statistically significant learning experience. Therefore, *HMM-Glasses* currently learns the probabilities of the part texts (i.e. words) in the question. This leaves the model with no dependency relation information at hand. However, it is compensated by the *Distiller* as the experts use by definition only the dependency rules for extraction.

Combination of the Distiller and HMM-Glasses. Recall that the *Distiller* outputs the focus parts with a single total confidence score of the expert that produced the results. In addition with the part-wise confidences that *HMM-Glasses* produces, we have:

$$\begin{array}{cc} \text{HMM} & \text{Distiller} \\ \left\{\begin{array}{l} \langle \text{fpn}_1, fpt_1, fpc_1 \rangle \\ \langle \text{fpn}_2, fpt_2, fpc_2 \rangle \\ \vdots \\ \langle \text{fpn}_p, fpt_p, fpc_p \rangle \end{array}\right. & \left.\begin{array}{l} \langle \text{fpn}_1, fpt_1, fpc \rangle \\ \langle \text{fpn}_2, fpt_2, fpc \rangle \\ \vdots \\ \langle \text{fpn}_q, fpt_q, fpc \rangle \end{array}\right\} \end{array}$$

Combination of the candidate focus parts produced by different models is performed in a part-wise manner. In other words, models try to convince each other about each part being among the final focus parts. To do this, we make use of the fpc scores, weight them with the models' individual f-scores over the training data and grab the maximum. Note that, if a part is determined as a candidate focus part by only one of the models M_1 (i.e. the other model M_2 predicts that this part is not a focus part), then we compute the confidence score of M_1 as described above and compare it with the f-score of M_2. If the confidence score of M_1 is greater than that of M_2, the word is classified as a focus part, otherwise it is excluded from the focus.

4.2 Class Extraction

For question classification, we manually pre-determined two types of classes, namely coarse and fine classes, adapted from [12,13], with different semantic resolutions. A question's fine class establishes a strong link to the specific domain at hand, while its coarse class essentially introduces a generality into the model that would render the classification applicable in domains other than Geography.

Currently we have seven coarse classes (see Table 2), along with a total of 57 fine classes. In this study, we only concentrated on coarse classes. We plan to perform classification of fine classes using statistical approaches, which requires comprehensive number of questions in each fine class.

Table 2. Coarse Classes for the Geography Domain

Question Class	*Frequency (%)*
DESCRIPTION	25.2
NUMERIC	24.2
ENTITY	19.6
TEMPORAL	12.4
LOCATION	11.9
ABBREVIATION	3.8
HUMAN	2.4

In order to classify a given question into one of the coarse classes, we devised a set of common phrases for each class unique to that class. For example, for the class NUMERIC, we have two phrases: "kaç"*(how many/how much)* and "kadardır"*(this much/that many)*. The classifier searches for these patterns in a given question and classifies accordingly.

We additionally implement a statistical classifier that employs a tf-idf based weighted bag-of-words strategy, as a baseline model to compare with the rule-based approach. In baseline model the weight of a word w for a class c is computed as follows.

$$\text{tf-idf}_{w,c} = tf_{w,c} \times idf_w$$

where $\text{tf}_{w,c}$ indicates the number of times word w occurs in class c, and idf_w is computed as shown below.

$$idf_w = \log \frac{\text{\# of classes}}{\text{\# of classes containing w}}$$

Then, for a given question Q, we assign it to the class that maximizes the sum of the tf-idf scores of the words in the question:

$$\arg\max_c \sum_{w \in Q} \text{tf-idf}_{w,c}$$

5 Data

One of the major contributions of this study is to provide a gold standard, diverse set of Turkish questions from the prototype domain of Geography, manually annotated by human experts. The data set contains 977 instances in the following format: {QuestionText | FocusPartTexts | CoarseClass | FineClass }.

Approximately 30 percent of the dataset consisted of actual questions posed by teachers, collected from Geography-related textbooks and online materials. The rest

were generated by three of the researchers, who are educational technologists, based on actual Geography texts used in grades 9 – 12 in high schools in Turkey.

Inter-Annotator Agreement. We made use of two strategies in data annotation: focus annotation and QClass annotation. Three researchers (two of whom are educational technologists) manually identified the focus in each question, while two researchers (one educational technologist) annotated the questions for QClass. The evaluations were later compared to the developer's judgment. The inter-annotator agreement scores for focus was 82%, and for QClass was 92%.

6 Evaluation and Results

One of the major challenges we face was not having a suitable baseline (from previous studies etc.) to indicate the actual hardness of the problem and the actual efficiency of our solutions. Therefore, we implemented a baseline model for focus extraction that identifies the words adjacent to a question keyword for certain proximity as focus parts. The proximity model has slightly worse than, but similar results with the tf.idf model. We chose to include only the baseline with the best results (i.e. tf.idf) for a clear comparison. Note that the baseline models are intentionally designed to be rather simple, because there is no prior study on statistical question analysis on Turkish. Therefore, the baselines are kept simple in order to set the lower bounds of the problem. Moreover, a tf-idf based statistical baseline model that employs a bag-of-words strategy is implemented for question classification as well. All the results are reported as comparisons to these baseline models in Table 3 and Table 4.

Table 3. Evaluation Results of All Models for Focus Extraction

Model	*Precision*	*Recall*	*F-Score*
Baseline (tf.idf model)	**0.769**	**0.197**	**0.290**
Distiller (Generic Enabled)	0.714	0.751	0.732
Distiller (Generic Disabled)	0.816	0.623	0.706
HMM-Glasses (Backward Mode)	0.839	0.443	0.580
HMM-Glasses (Forward Mode)	0.847	0.495	0.625
HMM-Glasses (Forward and Backward Mode)	0.821	0.515	0.633
Combined (Generic Enabled, Backward)	0.734	0.841	0.784
Combined (Generic Enabled, Forward)	0.732	0.846	0.785
Combined (Generic Enabled, Forward & Backward)	**0.721**	**0.851**	**0.781**
Combined (Generic Disabled, Backward)	0.821	0.759	0.789
Combined (Generic Disabled, Forward)	0.818	0.765	0.791
Combined (Generic Disabled, Forward & Backward)	**0.802**	**0.776**	**0.788**

Table 4. QClass Classification Results. Upper section is baseline tf-idf based model, and lower section is rule-based model.

Classses	*Precision*	*Recall*	*F-Score*
Description	0.662	0.908	0.764
Temporal	0.767	0.618	0.670
Numeric	0.801	0.758	0.776
Entity	0.100	0.025	0.040
Abbreviation	0.933	0.766	0.823
Location	0.759	0.212	0.312
Human	0.600	0.600	0.600
Tf.Idf Overall	**0.660**	**0.555**	**0.569**
Description	0.874	0.732	0.797
Temporal	1.000	1.000	1.000
Numeric	0.995	0.911	0.951
Entity	0.603	0.817	0.694
Abbreviation	0.871	0.894	0.883
Location	0.944	0.880	0.911
Human	0.869	0.833	0.851
Rule-based Overall	**0.879**	**0.867**	**0.869**

Since the data on which our models are evaluated have been prepared in this course of study, we build our strategy of evaluation around the concept of hygene, where we ensure two fundamental principles. Firstly, at any point and for each model, scores are obtained from the results produced for questions with which the model *never* crossed before. Secondly, for a reasonable comparison between the models, same scores are computed for different models with different settings using the *same* questions at each iteration of the evaluation.

To evaluate the *Distiller*, the rule-based experts are developed by using only the first 107 questions, that we had at the beginning. Therefore, the remaining questions are safely treated as test data, as there were no modifications done after having a larger number of questions.

Evaluations for all the models are performed using stratified 10-fold cross-validation over all the questions. The final results (i.e. precision, recall and f-score) for focus extraction are obtained by macro-averaging the individual results.

Recall that the *Distiller* has the option to enable and disable the generic expert, while the *HMM-Glasses* has *forward*, *backward* and *forward & backward* modes that calibrate the serialization of the dependency tree. All the different combinations of these settings for each model are seperately evaluated both individually and in combination, in each iteration of the folding process. The results for focus extraction and question classification are shown in Tables 3 and 4, respectively.

6.1 Focus Extraction Results

Individual evaluation of the *Distiller* resulted in comparable precision scores along with lower recall scores (compared to the combined models). A noticable outcome of the

Distiller evaluations is the behavior of the generic expert. Results indicate that generic expert lowers the accuracy of the retrieved results (i.e. precision), while increasing the coverage (i.e. recall) of the model. However, the two effects do not compensate, as the results show that f-score of the *Distiller* with the generic expert enabled is higher than the one with the generic expert disabled.

Distinct evaluation of the effect of the serialization methods indicates that for forward and backward modes, the forward mode is slightly better than the backward mode considering the f-scores. Backward mode seems to increase the recall of any model to which it is included, however, f-scores indicate that this increase in recall is not useful, because it in fact lowers the performance of the combined models whenever it is included.

In general, although the individual accuracies of the models are reasonable enough, the increase in the coverage (recall) for all combined models, having both the *Distiller* and *HMM-Glasses*, compared to the individual recall scores indicate that the combination is useful, as it does not sacrifice the precision scores that we observe in individual evaluations, thereby increasing also the f-scores. Therefore, we can conclude that the models complement each other nicely.

6.2 ClassRules Results

Results show that exploiting the domain knowledge resulted in a significant success that a statistical baseline model could not get near. However, manually crafted set of rules are a big problem when changing the domain. Therefore, a statistical learner that will automatically learn these domain specific phrases is planned for further development, since it requires significant amount of instances for each class. This scarcity is also the reason we leave the identification of fine classes for a future study. Table 4 shows the macro-averaged precision, recall and f-score of coarse class identification of the rule-based classifier, along with the results of the tf-idf based baseline classification.

7 Conclusion

In this study, we presented a novel combination of rule-based and statistical approaches to question analysis, employed in a closed-domain question answering system for an agglutinative language, such as Turkish. Our question analysis consists of focus extraction and question classification. For focus extraction, we have multiple rule-based experts for most frequent question types in Turkish. Additionally, we described a HMM-based novel sequence classification approach for focus extraction, along with combining the results of both rule-based and statistical models according to the individual confidence scores of each model. For question classification, we employed a rule-based classifier which uses pattern phrases uniqe to each class. We implemented baseline models for both problems, and have reported here the comparisons. In addition to the methodology offered, we also provide a set of manually annotated questions for both reproducability and further research.

Acknowledgments. This work was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) under grant number 113E036.

References

1. Allam, A.M.N., Haggag, M.H.: The question answering systems: A survey. International Journal of Research and Reviews in Information Sciences (IJRRIS) 2 (2012)
2. Benoit, D., Demaine, E.D., Munro, J.I., Raman, V.: Representing trees of higher degree. In: Dehne, F., Gupta, A., Sack, J.-R., Tamassia, R. (eds.) WADS 1999. LNCS, vol. 1663, pp. 169–180. Springer, Heidelberg (1999)
3. Bunescu, R., Huang, Y.: Towards a general model of answer typing: Question focus identification. In: International Conference on Intelligent Text Processing and Computational Linguistics (CICLING) (2010)
4. Dominguez-Sal, D., Surdeanu, M.: A machine learning approach for factoid question answering. Procesamiento de Lenguaje Natural (2006)
5. Er, N.P., Çiçekli: A factoid question answering system using answer pattern matching. In: International Joint Coneference on Natural Langauge Processing, pp. 854–858 (2013)
6. Eryiğit, G.: The impact of automatic morphological analysis & disambiguation on dependency parsing of turkish. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC), Istanbul, Turkey (2012)
7. Eryiğit, G., Nivre, J., Oflazer, K.: Dependency parsing of turkish. Computational Linguistics 34, 357–389 (2008)
8. Ferrucci, D.A.: Introduction to "this is watson". IBM Journal of Research and Development 56, 1–15 (2012)
9. Gupta, P., Gupta, V.: A survey of text question answering techniques. International Journal of Computer Applications 53, 1–8 (2012)
10. Katz, B.: Annotating the world wide web using natural language. In: Proceedings of the 5th RIAO Conference on Computer Assisted Information Searching on the Internet, pp. 136–159 (1997)
11. Lally, A., Prager, J.M., McCord, M.C., Boguraev, B.K., Patwardhan, S., Fan, J., Fodor, P., Chu-Caroll, J.: Question analysis: How watson reads a clue. IBM Journal of Research and Development 56, 2:1–14 (2012)
12. Li, X., Roth, D.: Learning question classifiers: the role of semantic information. Natural Language Engineering 12, 229–249 (2006)
13. Metzler, D., Croft, B.W.: Analysis of statistical question classification for fact-based questions. Information Retrieval 8, 481–504 (2005)
14. Munro, J.I., Raman, V.: Succinct representation of balanced parentheses and static trees. SIAM J. Comput. 31, 762–776 (2002)
15. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryiğit, G., Kübler, S., Marinov, S., Marsi, E.: Maltparser: A language-independent system for data-driven dependency parsing. Natural Language Engineering Journal 13, 99–135 (2007)
16. Şahin, M., Sulubacak, U., Eryiğit, G.: Redefinition of turkish morphology using flag diacritics. In: Proceedings of The Tenth Symposium on Natural Language Processing (SNLP 2013) (2013)
17. Viterbi, A.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory 13 (1967)
18. Wen, L., Amagasa, T., Kitagawa, H.: An approach for XML similarity join using tree serialization. In: Haritsa, J.R., Kotagiri, R., Pudi, V. (eds.) DASFAA 2008. LNCS, vol. 4947, pp. 562–570. Springer, Heidelberg (2008)
19. Zheng, Z.: Answerbus question answering system. In: Proceedings of the Second International Conference on Human Language Technology Research (HLT), pp. 399–404 (2002)