University of Hagen at CLEF 2008: Answer Validation Exercise

Ingo Glöckner
Intelligent Information and Communication Systems (IICS),
FernUniversität in Hagen, 58084 Hagen, Germany
iglockner@fernuni-hagen.de

Abstract

RAVE (Real-time Answer Validation Engine) is a logic-based answer validator/selector designed for application in real-time question answering. RAVE uses the same toolchain for deep linguistic analysis and the same background knowledge as its predecessor (MAVE), which took part in the AVE 2007. However, a full logical answer check as in MAVE was not considered suitable for real-time answer validation since it requires parsing of all answer candidates. Therefore RAVE uses a simplified validation model where the prover only checks if the support passage contains a correct answer at all. This move from logic-based answer validation to logical validation of supporting snippets permits RAVE to avoid any parsing of answers, i.e. the system only needs a parse of the question and pre-computed snippet analyses. In this way very low validation/selection times can be achieved. Machine learning is used for assigning local validation scores using both logic-based and shallow features. The resulting local validation scores are improved by aggregation. One of the key features of RAVE is its innovative aggregation model, which is robust against duplicated information in the support passages. In this model, the effect of aggregation is controlled by the lexical diversity of the support passages for a given answer. If the support passages have no terms in common, then the aggregation has maximal effect and the passages are treated as providing independent evidence. Repetition of a support passage, by contrast, has no effect on the results of aggregation at all. In order to obtain a richer basis for aggregation, an active validation approach was chosen, i.e. the original pool of support passages in the AVE 2008 test set was enhanced by retrieving additional support passages from the CLEF corpora. This technique already proved effective in the AVE 2007. The development of RAVE is not finished yet, but the system already achieved an F-score of 0.39 and a selection rate of 0.61 compared to optimal selection. Judging from last year's runs of MAVE (with a 0.93 selection rate and F-score of 0.72), this may look disappointing. However, the AVE task for German was much more difficult this year, and the F-score gain of RAVE (over the 100% yes baseline) and qa-accuracy gain (compared to random selection) are better than in last year's runs of MAVE.[1]

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering, Selection process*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Predicate Logic, Semantic networks*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing

General Terms

Experimentation, Measurement, Verification

Keywords

Logical Answer Validation, Answer Selection, Question Answering, Recognizing Textual Entailment (RTE), Information Fusion, Robust Inference

[1]Funding by the DFG (Deutsche Forschungsgemeinschaft) under contract HE 2847/10-1 (LogAnswer) is gratefully acknowledged.

1 Introduction

Answer validation for question answering (QA) systems is often construed as a problem of recognizing textual entailment (RTE). In this approach, the question (e.g. *'When was the Eiffel tower constructed?'*) and the answer candidate to be validated (e.g. *'1889'*) are turned into a textual hypothesis (*'The Eiffel tower was constructed in 1889'*) which is then checked against a supporting snippet extracted from the document collection – a task which can be implemented as a logical entailment test. The starting point for the current work is MAVE [2], an answer validator for German which embarks on this logic-based approach. While MAVE achieved excellent results in the AVE 2007, the system was not yet suitable for answer validation in interactive QA. When used for real-time question answering, the validator must be able to evaluate hundreds of candidate answers in a few seconds. To achieve this, a departure from the RTE approach is inevitable. The reason is that there is no time available to construct a logical representation for the answer candidates, i.e. syntactic-semantic parsing of hundreds of answers candidates (or hundreds of textual hypotheses constructed from question and answers) at query time is not feasible for real-time QA. There is sufficient time for parsing the question, however (since only one question must be analyzed for each query of a user), and there is also sufficient time for parsing all documents and possible support passages (since the corpus is known in advance, so the linguistic analysis of all documents in the corpus can be done at indexing time). Based on these observations, a new approach based on logical passage validation was proposed [3]. The basic idea is that of limiting the logical validation to a logical verification of the provided snippets: the system does not form a hypothesis from question and answer, but only tries to prove the logical representation of the question from the logical representation of the support passage. In this way the system can determine if the snippet contains a correct answer at all (not necessarily identical to the answer candidate). If the snippet does not contain a correct answer, then the snippet does not provide evidence for *any* answer candidate extracted from it, and the answer-support pair can be rejected on logical grounds. If the considered snippet does contain a correct answer, however, then additional (extra-logical) criteria are needed for verifying that a given answer candidate is indeed the correct answer contained in the passage. By avoiding the parsing of answers, the current validator, RAVE, can meet the requirements on processing speeds: an exact logical validation for 200 retrieved passages is accomplished in 2.4 seconds on average [3]. There is also an anytime variant of the method which allows the system to generate validated answers with a user-specified maximum latency [1, 7].

Another important improvement of RAVE compared with its predecessor is the treatment of replicated information in aggregation. This is perhaps not so important for the Wikipedia corpus (which contains few redundancy) but it is crucial when working with news corpora (that may contain several duplicates and variants of the same news) and when setting up a multi-stream system since the same answer candidate together with the same supporting snippets can occur in several QA streams. The problem that a simple aggregation model faces here is 'spurious' aggregated evidence, i.e. repetition of a support passage (or of minor variants) could result in an unwanted increase of the confidence score. To eliminate this problem, a replication-tolerant aggregation model was designed for RAVE. In this model, repetition of an answer-support passage pair does not affect aggregation results at all.[2] On the other hand, the effect of aggregation is maximal when the aggregated supporting snippets have no terms in common.

The paper is organized as follows: Sect. 2 explains the system architecture of RAVE. The focus is placed on the validation core; for details on the actual use of RAVE in QA systems see [1, 7]. Sect. 3 presents the results of RAVE in the AVE 2008, together with ablation studies which reveal the effect of various system components. The paper closes with a discussion of the progress made and also sketches necessary improvements.

2 System description

2.1 Overview

The architecture of the RAVE system is shown in Fig. 1. The input to the system comprises a question together with answer candidates for the question and the supporting text snippets. In order to introduce

[2]To be precise, the contribution of repeated answer-snippet pairs is the maximum score of any of these items; see Sect. 2.9

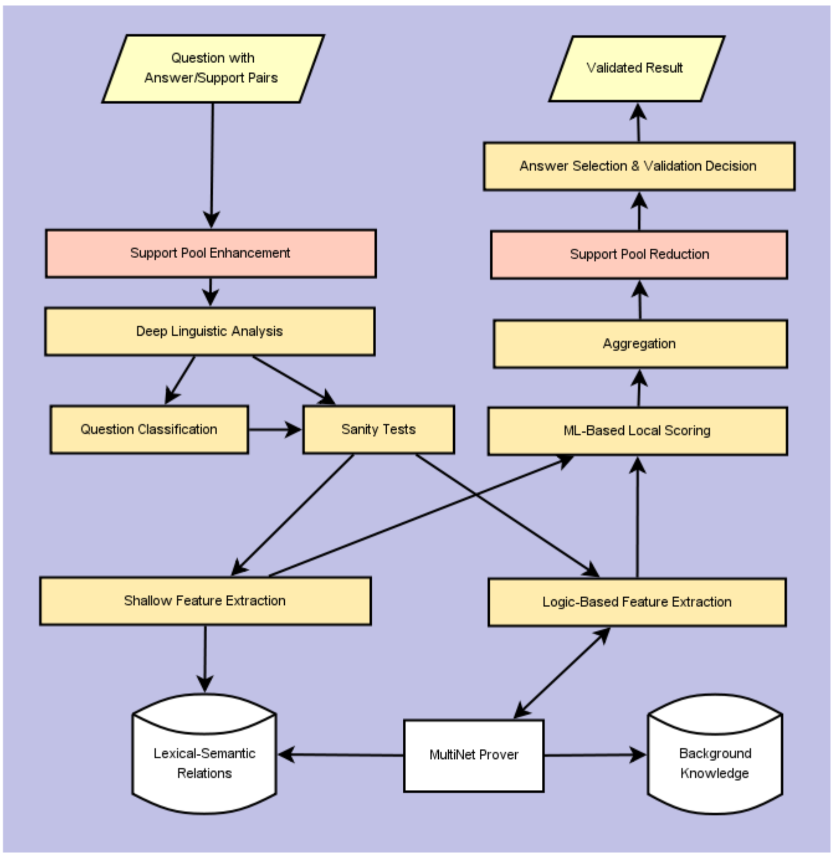

Figure 1: Architecture of the RAVE system

more redundancy for aggregation, a first step of *support pool enhancement* is used which retrieves additional supporting passages for the candidate answers from the document collections. Notice that this step is not part of the core system since there is usually sufficient redundancy in the QA streams. The question is subjected to a *deep linguistic analysis*. In the AVE, parsing the supporting snippets during validation was also necessary due to the presence of illegal document ids and corrupted snippets in the test set. In regular operation, however, RAVE never parses any snippets at query time, since the deep parse of the snippet can always be fetched from the pre-analyzed document collections. The *question classification* serves to identify the descriptive core of the question, and it also determines the expected answer type (EAT) and the question category. Depending on the question classification, the system performs a number of *sanity tests* on the answer candidates which eliminate trivial answers, for example. The remaining answers are validated based on the results of *shallow feature extraction* (like lexical overlap) and *logic-based feature extraction*. A local score is then computed by applying a classifier obtained by machine learning. Aggregation is used to determine a combined score for each answer which captures the joint evidence of all snippets supporting the answer (including the retrieved, auxiliary snippets introduced by the support pool enhancement). After aggregation, these auxiliary support passages are eliminated in the *support pool reduction* step since they have served their purpose of refining answer evidence. The final step then involves the selection of the best answer based on the aggregated evidence for the answer and the justification strengh of the considered validation item in the test set. The remaining answers are classified as validated or rejected depending on a given threshold.

In formal terms, the AVE task to be solved by RAVE can be described as follows. The AVE test set consists of validation items $i \in \mathcal{I}$ given by the question string q_i, the answer candidate a_i and the supporting snippet s_i. For convencience, let $\mathcal{Q} = \{q_i : i \in \mathcal{I}\}$ denote the set of all questions in the test set and let

$\mathcal{I}_q = \{i \in \mathcal{I} : q_i = q\}$ be the set of all validation items for a given question $q \in \mathcal{Q}$.

The goal of the answer validation and selection task is that of assigning a validation decision $v_i \in \{\textit{REJECTED}, \textit{SELECTED}, \textit{VALIDATED}\}$ and a confidence value $c_i \in [0, 1]$ to each $i \in \mathcal{I}$ such that at most one answer for each question is selected, i.e. $|\{i \in \mathcal{I}_q : v_i = \textit{SELECTED}\}| \leq 1$. Moreover answers can only be validated if an answer has been selected as the best answer, i.e. if $\{i \in \mathcal{I}_q : v_i = \textit{SELECTED}\} = \varnothing$, then $\{i \in \mathcal{I}_q : v_i = \textit{VALIDATED}\} = \varnothing$ as well.

2.2 Support Pool Enhancement

As in the last year, the AVE 2008 test set has also been constructed in such a way that every answer candidate occurs only once, i.e. there is virtually no redundancy available in the test set. Typical answer selection techniques rely heavily on redundancy, though, since the existence of many support passages for a given answer often indicates that the answer is correct. In a logic-based answer validation and selection setting, redundancy also helps increase robustness: If several snippets are available which support the same answer, then there is a better chance that a deep linguistic analysis exists for at least one of the support items for an answer. On the other hand, if there is only one support item for the answer (i.e. no redundancy) and if parsing of the support item fails, then the prover cannot be applied for logical validation. Thus, it makes sense to search for additional supporting snippets for each answer in the document collections, and add these snippets to the support pool as auxiliary validation items.[3] Thus, RAVE starts from the pool of 'regular' or 'original' validation items $i \in \mathcal{I}_q$ for a question q. Then $A_q = \{\alpha_i : q_i = q\}$ is the set of answer candidates q in the test set. An existing QA environment – in this case the IRSAW system [7] – is used to actively search for additional supporting snippets for each of the answer candidates in A_q. These are filtered from the set of answer/support pairs generated by the IRSAW QA streams for the question. This process results in a number of new support items for the answers in A_q. In order to improve recall, answers are clustered into groups of minor variants with the same 'answer key' $\kappa(a_i)$ by applying a simplification function κ on the answers.[4] Not only exact matches of answer candidates pass the filter, but also those which only share the same answer key. This process results in a set of auxiliary validation items $i \in \mathcal{I}'_q$ with $q_i = q$ and $\kappa(a_i) = \kappa(a_j)$ for some original answer $a_j \in A_q$, and a supporting snippet s_i for a_i found by the IRSAW QA system for the considered question. The original and auxiliary validation items are joined into the enhanced validation pool $\mathcal{I}^*_q = \mathcal{I}_q \cup \mathcal{I}'_q$ for q that contains all original and auxiliary support passages for the answers to be validated. Notice that exact duplicates are not included into $\mathcal{I}'_q$, i.e. each combination of answer key and supporting snippet is added only once.[5] Though RAVE does not use support pool enhancement when applied in actual QA systems, its discernment of 'regular' support items (which may be shown to the user) and 'auxiliary' support items (which only serve to improve selection, but may not be shown to the user) can be very useful in practice. One obvious application is the use of auxiliary document collections whose licensing conditions do not permit snippets from these collections to be shown to third parties. When treated as a source of 'inofficial' support items that are only internally used for deciding which answer candidates are the correct ones, such a corpus with rigid licensing conditions can still prove useful. Another obvious scenario is multilingual QA. In this case, considering snippets in a language that the user does not understand can still be valuable for improving results by aggregation. However, presenting such snippets to the user would be pointless.

2.3 Deep Linguistic Analysis

RAVE uses WOCADI [6], a syntactic-semantic parser for German, for computing deep linguistic analyses of the question and of the documents. As mentioned above, parsing of the snippets at validation time is normally not needed, since all documents are known in advance and can be pre-analyzed. A full parse is found for about 60% of the sentences in the CLEF corpora; in this case a semantic representation in the MultiNet formalism [8] is constructed which forms the basis for logical validation. If a full parse fails, then WOCADI still produces results of a morpho-lexical analysis (lemmata, possible word senses, numerals,

[3] Such an 'active validation approach' already proved useful for MAVE in the AVE 2007.

[4] The chosen simplification function [4] drops accents, removes whitespace and eliminates insignificant words from the answers. For example, $\kappa(\textit{im Jahr 2001}) = 2001 = \kappa(\textit{2001})$.

[5] Thus for all $i, j \in \mathcal{I}^*_q$, if $\kappa(a_i) = \kappa(a_j)$ and $s_i = s_j$, then $i = j$.

named entity types, results of compound decomposition) which can be utilized for implementing a fallback validation method that works for arbitrary sentences.

2.4 Question Classification

A rule-based approach is used for question classification, with a total of currently 127 classification rules. Consider the query *Nenne mir drei Beispiele für Vulkaninseln!* (*'Name three examples of volcanic islands!'*), for example. The classification rules of RAVE then determine the expected answer type (EAT; in this case: *island-name*), the query category (*factual*), the desired number of results (three) and the descriptive core of the query (*Vulkaninseln*, i.e. volcanic islands). The rules also remove parts of the query which are not part of the descriptive question core. In this case, *Nenne drei Beispiele* (*'Name three examples'*) will be removed from the logical representation of the query and also from the set of word senses and numbers used for the lexical overlap test.

2.5 Sanity Tests

A number of sanity tests are applied to the answers and support passages in order to eliminate false positives. The MAVE system used 'soft' sanity checks (which merely reduce the confidence score of an answer candidate); these checks were applied late (after aggregation) in order to avoid the effect of a failed sanity check to be evened out by aggregation. RAVE, by contrast, uses strict sanity checks which are applied at the very moment that an answer or passage enters into the system. In this case, aggregation cannot compromise the effect of a failed sanity check because validation items which fail one of the tests are discarded immediately. The RAVE prototype currently uses the following sanity checks:

- A test for *trivial answers*, which checks if the answer merely repeats content of the question. Example: *Was ist die Arche Noah?* (*'What is Noah's Ark?'*) *– Die Arche Noah* (*'Noah's Ark'*).
- A test for *non-informative definitions*, which eliminates incomplete answers to definition questions involving relational nouns or nominalizations. Examples: *Wer war Antonio Gaudi?* (*'Who was Antonio Gaudi?'*) *– Gegenspieler* (*'opponent'*); *Was ist ein Echolot?* (*'What is Sonar?'*) *– zur Detektion* (*'for detection'*).
- A test for *failure of temporal restrictions*, currently implemented as a simple year check on support passages. Example: *Wer war Russlands Verteidigungsminister 1994?* (*'Who was Russia's Minister of Defence in 1994?'*) – rejected snippet: *Russlands Verteidigungsminister Gratschow zu Besuch in Bonn* (*'Russia's Minister of Defence, Gratschow, is visiting Bonn'*).
- Two tests for *incompatible measurement units*, which apply only to MEASURE questions:
  - The question requests a certain class of measurement units (e.g. length units) by naming the measurement dimension (*'What length…'*, *'How long…'*), but the answer contains an incompatible measurement unit. Example: *Bei welcher Temperatur schmilzt Eisen?* (*'At which temperature does iron glow?'*) *– 1,86 m* (*'1.86 m'*).
  - The question requests a specific measurement unit, but the measurement unit in the answer differs. Example: *Mit wie viel Dollar ist der UNESCO-Friedenspreis dotiert?* (*'How many Dollars is the UNESCO Prize for Peace endowed with?'*) *– 1 Million DM*.

2.6 Shallow Feature Extraction

RAVE extracts 17 shallow features which only depend on the results of morpho-lexical analysis and on the question classification. Therefore these shallow features can be computed for arbitrary snippets.

The following *passage features* only depend on the question and on the support passage (i.e. the need not be recomputed for each answer candidate):

- *failedMatch* Number of lexical concepts and numerals in the question which cannot be matched with the candidate document.

- *matchRatio* Relative proportion of lexical concepts and numerals in the question which find a match in the candidate document.
- *failedNames* Proper names mentioned in the question, but not in the passage.
- *containsBrackets* Indicates that the passage contains a pair of parentheses.
- *knownEat* Indicates that the expected answer type is known, i.e. question classification has succeeded.
- *testableEat* Indicates that the expected answer type is fully supported by the current implementation of the answer type check.
- *eatFound* Indicates that an occurrence of the expected answer type has been found in the snippet.
- *isDefQuestion* Indicates that the question is a definition question.
- *defLevel* Indicates that the snippet contains a defining verb or apposition ($defLevel = 2$) or a relative clause ($defLevel = 1$); else $defLevel = 0$.

The matching technique used to obtain the values of these shallow features also takes into account synonyms and nominalizations [2]. The *containsBrackets* feature was introduced since the queried information is often contained in parentheses, e.g. *Mount Everest (4,848 m)*. Thus the presence of brackets in the passage affects the relevance judgement.

The above answer-type related passage features are independent of any answer candidate. They describe aspects of the question classification and check for the presence of an expression with the expected answer type in the passage. The implementation of the answer type check in RAVE is not complete yet. While all answer types can be extracted from a successful parse of the snippet, the system is currently only able to extract proper name types (i.e., types of actual named entities) from answers and from snippets with a failed parse. The feature *testableEat* indicates if the expected answer type can always be checked (i.e. if the EAT is a named entity type), or if the answer type check is only possible for a support passage with known parse.

When using the validator in actual QA systems, an additional *irScore* feature is used which contains the original retrieval score of the support passage assigned by the retrieval system. Incorporating a retrieval score based on the tf/idf measure has a positive effect (this is known from tests on the training data), but in the AVE test set, the feature is of course not available.

In addition to the passage-related features, RAVE also uses several answer-related features which depend on the answer candidate. These features must be computed for each considered answer/passage combination.

- *awFailedMatch* Number of lexical concepts and numerals in the answer which cannot be matched with the snippet sentence.
- *awMatchRatio* Relative proportion of lexical concepts and numerals in the answer which find a match in the snippet sentence.
- *awFailedNames* Proper names mentioned in the answer, but not in the snippet sentence.
- *synthFailedMatch* Number of lexical concepts and numerals in the question or in the answer which cannot be matched with the snippet sentence.
- *synthMatchRatio* Relative proportion of lexical concepts and numerals in the question or in the answer which find a match in the candidate document.
- *synthFailedNames* Proper names mentioned in the question or in the answer, but not in the snippet sentence.
- *awLength* Length of the answer, i.e. the number of characters.
- *awEatMatch* Indicates that the actual answer type of the answer matches the expected answer type.

When using the validator in actual QA systems, there is also a *producerScore* feature which represents the relevance score assigned by the answer source that generated the answer candidate. RAVE then uses separate models for each stream in order to account for individual characteristics of each QA source. This customization is not possible in the AVE task since the QA sources and their characteristics are not known.

2.7 Logic-Based Feature Extraction

In order to obtain a more reliable basis for validation, RAVE also computes a set of logic-based features, given that parsing of question and snippet has succeeded. The basic idea of the logical passage validation approach is that proving the question from the supporting snippet can reveal whether the snippet contains a correct answer at all. RAVE thus attempts to prove the logical representation of the question from the logical representation of the supporting snippet and from its background knowledge.[6] The logical query is represented as a conjunction of literals. Consider the query *Nennen Sie einige einfache Elementarteilchen.* (*'Please name some elementary particles'*). The logical query which results from parsing and subsequent question analysis is: $\mathrm{prop}(FOCUS, \textit{einfach.1.1}) \wedge \mathrm{sub}(FOCUS, \textit{elementarpartikel.1.1})$. The example demonstrates how synonyms are used for normalization of word senses: *elementarteilchen.1.1* (elementary particle) is replaced by the canonical *elementarpartikel.1.1*. The FOCUS variable represents the queried information. If a proof of the query succeeds, then the FOCUS variable gets bound to an entity in the logical representation of the snippet. The WOCADI parser provides word alignment information which can be used to recover the substring of the snippet which corresponds to the binding of the FOCUS variable. Due to the availability of alignment information, this logic-based answer extraction can be performed very quickly. It is used to define several extraction-related features. The basic idea behind these features is that the ability of the system to verbalize the answer binding might also have something to say about the relevance of the passage.

RAVE extracts useful information even in the case that a proof of the complete query fails. In this case the prover returns the longest proven fragment of the query along with the answer substitution determined from the proof of this fragment. RAVE then extracts the number of proven literals and other features from the results of this partial proof. The system also supports relaxation (i.e. a failed proof can be restarted after simplifying the query by skipping a problematic literal), but this mechanism was switched off in the AVE since it has high computational cost and only a modest effect on validation quality [5].

- *skippedLitsLb* Number of known failed literals. Since relaxation is switched off, this feature is 0 if a complete proof succeeds and 1 if a complete proof fails.
- *skippedLitsUb* Number of known failed literals, plus literals with unknown status. Since no relaxation is used, this feature equals the number of all literals minus the length of the longest proven fragment of the query.
- *litRatioLb* Fraction of actually proved literals compared to the total number of query literals, i.e. $1 - \textit{skippedLitsUb}/\textit{allLits}$.
- *litRatioUb* Fraction of potentially provable literals vs. all literals, i.e. $1 - \textit{skippedLitsLb}/\textit{allLits}$.
- *boundFocus* Indicates that a binding for the queried variable was found.
- *extractedFocus* Signals that RAVE has managed to extract the substring of the snippet which corresponds to the computed binding of the FOCUS variable.
- *npFocus* Indicates that the answer string obtained by the logical answer extraction method is a nominal phrase (NP).
- *focusEatMatch* Indicates that the answer type of the answer binding found by the prover matches the expected answer type.
- *focusDefLevel* Indicates that the answer binding found by the prover corresponds to an apposition (result: 2) or to an NP with a relative clause (result: 1), else 0.

[6]RAVE uses the same sources of background knowledge as the MAVE system described in [2].

It should be pointed out that all logic-based features depend only on the question and on the supporting snippet, i.e. one proof per snippet is sufficient to determine these features. No attempt is made at this time to compare the answer string extracted as a side-effect of logical passage validation with the answer candidates to be checked. There is apparent room for improvements here, and features which relate the answer candidates to the results of logical validation/extraction will be added in a later version of RAVE.

2.8 ML-Based Local Scoring

A machine learning approach is used on order to assign an evidence score $\eta_i \in [0, 1]$ to each $i \in \mathcal{I}_q^*$ which estimates the probability that answer a_i is both correct and properly supported by snippet s_i. A basic technique suitable for the task was developed in [3], using the Weka toolbench [10]. Bagging of decision trees is used for learning a model which then serves for obtaining probability estimates. Due to the unbalanced training data (there are typically few positive exemplars and a large number of negative exemplars), cost-sensitive learning by re-weighting of training examples was applied [10, p. 165]. In order to emphasize the results of interest (i.e. the correct answers), a weight of $\beta = 0.3$ for false positives and of 1.0 for false negatives was used throughout.[7] Class probabilities were obtained from the numbers of positive and negative examplars at the leaves of the decision trees in the usual way. However, as a result of the re-weighting of training examples, these numbers refer to the re-weighted training set and do not reflect the true class probabilities. This effect is corrected by applying the function $\rho(x) = \beta x/(1 - x + \beta x)$ to the relative frequencies of the YES class at the leaves of the decision trees. In particular, for $\beta = 0.3$, one obtains $\rho(0.5) \approx 0.23$ as the appropriate threshold for the validation decision. This basic approach was used for learning separate models for factual vs. definition questions and for the full set of features vs. shallow-only features. However, the AVE 2008 development set was considered too small for providing useful training data. Therefore existing training sets for the IRSAW system [7] were re-used as training sets for RAVE. These training sets cover a total of 10,447 annotated answer candidates and 27,919 passages generated in an IRSAW run on the CLEF 2007 questions.[8] Notice that all these models were trained on single-sentence snippets. For the moment, RAVE handles multi-sentence snippets by iterating over all sentences and choosing the maximum sentence score for η_i. RAVE also remembers the sentence that produced the maximum score; this best sentence is later needed for aggregation. Moreover there is a special treatment of full-sentence answers to definition questions. If a full-sentence answer to a definition question is recognized, and if the answer sentence is part of the snippet, then RAVE judges the correctness of the long answer by a classifier trained to recognize sentences which contain answers to definition questions. If the score obtained in this way exceeds the result obtained by the regular method determining the evaluation score, then the result of the full-sentence classifier is chosen for η_i.

2.9 Aggregation

So far, we have local scores η_i that express the direct evidence for an answer a_i judging from a particular supporting snippet s_i. Intuitively, the plausibility of an answer candidate increases when there are multiple support passages for the answer. In order to capture the joint evidence $\gamma(a)$ for an answer judging from all snippets supporting it, an aggregation technique is used. The aggregation model of RAVE was designed to be robust against replicated information. That is, multiple copies of the same supporting snippet should not result in an increase of the aggregated score since such copies do not provide independent evidence. Assuming that more diversity of the snippets usually means better independence of the sources, strongly overlapping snippets should also have few effect on the aggregated score, while snippets with no common terms at all should achieve the strongest gain of the aggregation result. These requirements can be met by the following aggregation model.

[7]The Weka Bagging learner was chosen for learning the classifier, using default settings (averaging over results of 10 decision trees computed by the Weka REPTree decision tree learner). It was wrapped in a Weka CostSensitiveClassifier configured for reweighting of training examples. The following Weka command line was used to generate the classifiers:

```
weka.classifiers.meta.CostSensitiveClassifier -cost-matrix "[0.0 0.3; 1.0 0.0]" -S 1 -W
weka.classifiers.meta.Bagging -- -P 100 -S 1 -I 10 -W weka.classifiers.trees.REPTree --
-M 2 -V 0.0010 -N 3 -S 1 -L -1
```

[8]An answer was annotated YES if the answer is correct and supported, and NO otherwise. All answer candidates were produced by the MIRA answer extractor of IRSAW which appears representative of mainstream QA technology.

We assume a simplification mapping κ from answers to simplified answer keys (see Sect. 2.2; in the simplest case, κ can also be the identity). The set of answer keys for a given question is $K_q = \{\kappa(a_i) : i \in \mathcal{I}_q\}$. For $k \in K_q$, let $\mathcal{I}_{q,k} = \{i \in \mathcal{I}_q : \kappa(a_i) = k\}$, i.e. $\mathcal{I}_{q,k}$ is the set of all support items for the considered answer key. For each $i \in \mathcal{I}_q$, let Ω_i be the set of term occurrences[9] in the supporting snippet s_i. For a term occurrence $\omega \in \Omega_i$, let $t(\omega)$ be the corresponding term. Further let $T_i = \{t(\omega) : \omega \in \Omega_i\}$ be the set of all terms in the support passage and $T_k = \bigcup \{T_i : i \in \mathcal{I}_{q,k}\}$ the set of all terms which occur in a passage for answer key $k \in K_q$. We abbreviate

$$\mu(k,t) = \min\left\{(1-\eta_i)^\nu : i \in \mathcal{I}_{q,k},\ t \in T_i\right\}, \quad \nu = \tfrac{\operatorname{occ}(t,i)}{|\Omega_i|}, \tag{1}$$

where $\operatorname{occ}(t,i) = |\{\omega \in \Omega_i : t(\omega) = t\}|$ is the occurrence count of term t in snippet i, and η_i is the correctness probability of the passage estimated by the ML classifier. The aggregated support for an answer key $k \in K_q$ is then given by

$$\gamma(k) = 1 - \prod_{t \in T_k} \mu(k,t)\,. \tag{2}$$

We extend this to extracted answers by stipulating that $\gamma(a) = \gamma(\kappa(a))$.

In general, the effect of aggregation will be strongest when two aggregated passages have no terms in common (i.e. the passages represent independent evidence), and there will be no effect of aggregation at all when the same passage is encountered repeatedly. The following properties of the method are obvious:

a. Suppose that $i, i' \in \mathcal{I}_q^*$ with $\kappa(a_i) = \kappa(a_{i'})$, $s_i = s_{i'}$ (i.e. the two snippets are identical and support equivalent answers). Further suppose without loss of generality that $\eta_i \geq \eta_{i'}$ (otherwise i and i' can be swapped).[10] Let $\gamma(a_i)$ be the aggregation result obtained from $\mathcal{I}_q^*$ and $\gamma'(a_i)$ be the aggregation result obtained from $\mathcal{I}_q^* \setminus \{i'\}$. Then $\gamma'(a_i) = \gamma(a_i)$, i.e. the aggregation result is not affected by the duplicate support item i. Moreover, when there are repeated snippets, then the maximum score of these support items will enter into the aggregation.

b. It generally holds that $\gamma^{\max}(k) \leq \gamma(k) \leq \gamma^{\text{indep}}(k)$, where

$$\gamma^{\max}(k) = \max\{\eta_i : i \in \mathcal{I}_{q,k}\} \tag{3}$$

$$\gamma^{\text{indep}}(k) = 1 - \prod_{i \in \mathcal{I}_{q,k}} (1-\eta_i)\,. \tag{4}$$

c. The equality $\gamma(k) = \gamma^{\text{indep}}(k)$ holds exactly if either $\eta_i = 1$ for some $i \in \mathcal{I}_{q,k}$, or if $T_i \cap T_j = \varnothing$ for all $i, j \in \mathcal{I}_{q,k}$ with $\eta_i > 0$ and $\eta_j > 0$. In particular, when there is no overlap between the snippets at all, then the evidence provided by the snippets is treated as independent.

The actual implementation of the method in RAVE is slighly more complicated because RAVE uses word senses and numerals as the terms (rather than word forms or word stems). The point is that the occurrence of a word only corresponds to a unique word sense if the WOCADI parser can determine a complete parse of the snippet. When a parse fails, however, then the possible word senses for a word cannot be disambiguated. The original approach is therefore changed in such a way that multiple word-sense alternatives for a given word occurrence can be handled as well. A word occurrence ω now corresponds to a finite set of terms $T(\omega) = \{t_1, \ldots, t_r\}$ with $r > 0$, as compared to the single term $t(\omega)$ in the original approach. If $T(\omega)$ contains more than one alternative, then all elements in $T(\omega)$ are considered as occurring with weight $1/r$. Now abbreviating $T_i = \bigcup\{T(\omega) : \omega \in \Omega_i\}$, we adapt the definition of the term occurrence count for $t \in T_i$ as follows,

$$\operatorname{occ}(t,i) = \sum_{\{\omega \in \Omega_i : t \in T(\omega)\}} 1/\left|T(\omega)\right|\,. \tag{5}$$

[9] A term occurrence is a pair (t,i) where t is a term and i is the position of the occurrence in the passage. In RAVE, word senses and numerals (rather than words or word stems) are used as the terms.

[10] If different QA streams with different ML-based models are used, the same answer/snippet pair can be assigned different scores η_i, $\eta_{i'}$ since the streams may differ in reliability.

Another modification of the basic method is concerned with the considered segment of the snippet. Rather than extracting word senses and numerals from the whole snippet, RAVE only considers a single, 'best' sentence, viz the sentence which attained the best score when determining η_i. This modification is motivated as follows: if the supporting snippets are longer than necessary, then the excess parts of the snippets can wrongly suggest diversity (because the text surrounding the relevant information is different) though the relevant sentence is in fact only duplicated. In order to avoid such negative effects of long snippets that also contain irrelevant sentences, a single most relevant sentence is picked from each snippet.

Notice that the actual implementation does not use equations (1) and (2) directly, but rather switches to logarithms in order to improve numerical stability. Thus, the following equations are used:

$$\mu'(k,t) = \min\left\{\nu \cdot \ln(1-\eta_i) : i \in \mathcal{I}_{q,k},\ t \in T_i\right\}, \quad \nu = \tfrac{\mathrm{occ}(t,i)}{|\Omega_i|}, \tag{6}$$

$$\gamma(k) = 1 - e^{\sum_{t\in T_k} \mu'(k,t)} \tag{7}$$

If η_i for some $i \in \mathcal{I}_{q,k}$, we stipulate that $\gamma(k) = 1$, as before.

2.10 Support Pool Reduction

After aggregation, the auxiliary supporting snippets are no longer needed. They are dropped to prevent them from showing up in the final output of RAVE. This is achieved by reducing the support pool from $\mathcal{I}_q^*$ to the original cases in $\mathcal{I}_q$ again.

2.11 Answer Selection and Validation Decision

The global answer score $\gamma(a)$ obtained by aggregation abstracts from individual support items. However, the validation/selection decisions apply to answer/snippet pairs (a_i, s_i). Therefore the final selection score σ_i should not only depend on the aggregated score $\gamma(a_i)$ but also take the justification strength η_i of the given snippet s_i into account. The submitted runs use the following formula for computing the σ_i score:

$$\sigma_i = \frac{\eta_i\,\gamma(a_i)}{\max\{\eta_j : j \in \mathcal{I}_q^*, \kappa(a_i) = \kappa(a_j)\}} \tag{8}$$

This means that the 'best' support item in $\mathcal{I}_q^*$ which supports a given answer a, i.e. $i^* \in \mathcal{I}_q^*$ with $a_{i^*} = a$ and $\eta_{i^*} = \max\{\eta_j : j \in \mathcal{I}_q^*, \kappa(a_i) = \kappa(a_j)\}$, is boosted to the aggregated score $\sigma_{i^*} = \gamma(a_{i^*})$. The formula shows some undesirable effects, though. For example, if the best support item for the considered answer a is an auxiliary support item $i^* \in \mathcal{I}_q'$ with a direct score $\eta_{i^*} = 1$, then the aggregation result $\gamma(a)$ has no effect at all, i.e. $\sigma_i = \eta_i$ for all $i \in \mathcal{I}_q^*$ with $\kappa(a_i) = \kappa(a)$. In order to avoid this effect, the current version of RAVE (after the AVE) now uses the following improved formula:

$$\sigma_i^{\mathrm{new}} = \frac{\eta_i\,\gamma(a_i)}{\max\{\eta_j : j \in \mathcal{I}_q, \kappa(a_i) = \kappa(a_j)\}} \tag{9}$$

where the index j ranges only over support items in the *original* set $\mathcal{I}_q$. The modified formula will boost the best original support item for a given answer, i.e. i.e. $i^* \in \mathcal{I}_q$ with $a_{i^*} = a$ and $\eta_{i^*} = \max\{\eta_j : j \in \mathcal{I}_q, \kappa(a_i) = \kappa(a_j)\}$, yielding the aggregated value $\sigma_{i^*} = \gamma(a)$.

Instead of (9), it is also possible to use a simple weighted average of aggregated and direct score, viz

$$\sigma_i^{(\lambda)} = \lambda\gamma(a_i) + (1-\lambda)\eta_i \qquad \text{for some } \gamma \in [0,1]. \tag{10}$$

Based on the assignment of final selection scores σ_i, the system determines a choice of $i^{\mathrm{opt}} \in \mathcal{I}_q$ which maximizes σ_i, i.e. $\sigma_{i^{\mathrm{opt}}} = \max\{\sigma_i : i \in \mathcal{I}_q\}$. The chosen i^{opt} is marked as $v_{i^{\mathrm{opt}}} =$ *SELECTED* if $\sigma_{i^{\mathrm{opt}}} \geq \theta_{\mathrm{sel}}$, where $\theta_{\mathrm{sel}} \in [0,1]$ is the selection threshold; otherwise i^{opt} is marked as $v_{i^{\mathrm{opt}}} =$ *REJECTED*. In particular, $\theta_{\mathrm{sel}} = 0$ can be used in order to force selection (i.e. the best validation item for a question will always be selected); this is usually a good choice for maximizing the number of correct selections. In experiments aiming at a good F-score, a threshold of $\theta_{\mathrm{sel}} = 0.23 \approx \rho(0.5)$ was used.

Table 1: Results of RAVE in the AVE 2008

model	f-score	f-gain	prec	p-gain	recall	qa-acc	qa-perf	sel-rate	s-gain
Run1	0.39	0.89	0.33	2.83	0.49	0.32	0.32	0.61	2.91
Run2	0.29	0.40	0.25	2.16	0.34	0.23	0.23	0.44	2.07

The non-best items $i \in \mathcal{I}_q \setminus \{i^{\mathrm{opt}}\}$ are classified as follows: if $v_{i^{\mathrm{opt}}} =$ *REJECTED*, i.e. no selection has been made, then $v_i =$ *REJECTED* for all $i \in \mathcal{I}_q \setminus \{i^{\mathrm{opt}}\}$ as well. On the other hand, if a selection has been made, i.e. $v_{i^{\mathrm{opt}}} =$ *SELECTED*, then we set $v_i =$ *VALIDATED* if $\sigma_i \geq \theta_{\mathrm{val}}$ and $v_i =$ *REJECTED* otherwise, where $\theta_{\mathrm{val}} \in [0, 1]$ is the decision threshold for validating the non-best items. In the experiments, $\theta_{\mathrm{val}} = 0.23 \approx \rho(0.5)$ was used throughout.

The confidence into this validation decision is given by $c_i = \sigma_i$ in the positive case that $v_i =$ *SELECTED* or $v_i =$ *VALIDATED*, and by $c_i = 1 - \sigma_i$ if $v_i =$ *REJECTED*.

3 Evaluation

Preliminaries

The AVE 2008 test set for German contains 1027 validation items for 119 questions. 111 of these answer/support items are classified as correct, 854 as wrong and 66 as undecided. A 100% YES baseline (all items accepted) thus achieves a precision of 0.12 and F-score of 0.21. A correct answer exists for 62 of the questions, i.e. the qa-accuracy (correct selections divided by number of questions) is bounded by 0.52. Random selection of an answer yields a qa-accuracy of 0.11 and selection rate (compared to optimal selection) of 0.21. These numbers are very different from the situation in 2007 where the test set for German had a precision of 0.25 and F-score of 0.40 for the 100% YES baseline, and a qa-accuracy of 0.28 and selection rate as high as 0.52 for random selection.

The support pool enhancement by mining for additional supporting snippets in the QA@CLEF corpora for German resulted in 579 auxiliary answer-snippet pairs and an enhanced pool with 1,606 items total.

Results of AVE 2008 Runs

The results of RAVE in the AVE 2008 are shown in Table 1. The following column labels are used: *f-score* (F-score, i.e. harmonic mean of precision and recall), *f-gain* (actual F-score divided by F-score for the 100% YES baseline), *prec* (precision for the YES class), *p-gain* (actual precision divided by precision of the 100% YES baseline), *recall* for the YES class, *qa-acc* (qa-accuracy, i.e. correct selections divided by number of questions), *qa-perf* (estimated qa-performance as defined in the AVE 2008 task description), *sel-rate* (selection rate, i.e. successful selections divided by optimal selections), *s-gain* (selection gain, i.e. actual selection rate divided by the selection rate for random selection). Both submitted runs used equation (8) for combining the local snippet score and the aggregated score. *Run1* also included the special treatment of full-sentence answers to definition questions. Since it was not clear from the QA@CLEF 2008 guidelines if such answers would be accepted or not, the second run was configured such that all full-sentence answers were rejected. However, the low scores for *Run2* demonstrate that defining sentences were generally accepted by the annotators. Therefore only *Run1*, whose treatment of defining sentences agrees with the official annotation, will be considered further in the following ablation experiments.

Ablation Studies and Results for System Variants

Table 2 lists the reference results for the current version of RAVE after correcting a few bugs that surfaced in the meantime.[11] The letter 'R' refers to the standard method of RAVE for combining local and aggregated

[11]The following bugs have been fixed: An error in the test for non-informative answers to definition questions resulted in many correct answers to be dropped. The lexical overlap test takes compound decompositions into account, but the implementation did not

Table 2: Reference runs for the ablation experiments: Boosting method

model	f-score	f-gain	prec	p-gain	recall	qa-acc	qa-perf	sel-rate	s-gain
RF	0.45	1.20	0.43	3.75	0.48	0.26	0.34	0.50	2.37
RQ	0.44	1.11	0.36	3.10	0.56	0.34	0.34	0.65	3.06

Table 3: Results for weighted averages instead of the boosting method

model	f-score	f-gain	prec	p-gain	recall	qa-acc	qa-perf	sel-rate	s-gain
WF0	0.37	0.78	0.40	3.44	0.34	0.21	0.28	0.40	1.91
WF.25	0.43	1.09	0.44	3.82	0.42	0.22	0.29	0.42	1.99
WF.5	0.43	1.10	0.43	3.76	0.43	0.23	0.30	0.44	2.07
WF.75	0.47	1.29	0.45	3.92	0.50	0.25	0.33	0.48	2.29
WF1	0.47	1.26	0.44	3.83	0.50	0.26	0.34	0.50	2.37
WQ0	0.36	0.75	0.31	2.69	0.43	0.29	0.29	0.56	2.68
WQ.25	0.40	0.95	0.34	2.93	0.50	0.29	0.29	0.55	2.60
WQ.5	0.41	0.99	0.34	2.99	0.51	0.30	0.30	0.58	2.75
WQ.75	0.45	1.18	0.37	3.20	0.58	0.33	0.33	0.63	2.98
WQ1	0.45	1.16	0.36	3.16	0.58	0.34	0.34	0.65	3.06

scores, see equation (9), 'F' means the use of $\theta_{\mathrm{sel}} = 0.23$ for F-score oriented runs, and 'Q' signals the use of $\theta_{\mathrm{sel}} = 0$ for runs aiming at qa-accuracy. The RQ method corresponds to that used for *Run1*.

We will first consider effects related to aggregation. Table 3 lists the results obtained when using equation (10) instead of (9) for a weighted average of the local score and the aggregated score with different values $\lambda \in \{0, 0.25, 0.5, 0.75, 1\}$ for the weight of the aggregated score. (This is symbolized by the letter 'W' and the value of λ shown as the suffix of the runs.) Choosing $\gamma = 0$ in the WF0 and WQ0 runs means that no aggregation is used at all. Comparing WF0 with RF, and WQ0 with WF, we find a strong effect of aggregation (plus 8 percent points of F-score and a similar increase in the selection rate). The best F-score of 0.47 is reached for $\lambda = 0.75$ and the best selection rate (of 0.65) for $\lambda = 1$. In this case, the selection and validation decision depends only on the aggregated evidence for the answer a_i, and the evidence η_i from the considered snippet s_i itself has zero weight. Surprisingly, this method still has relatively good F-score. The selection results for RF/WF1 and RQ/WQ1 are identical.

Table 4 shows the results obtained when replacing the replication-tolerant aggregation scheme of Sect. 2.9 with either 'best evidence' aggregation score $\gamma^{\max}$ given by (3), letter 'B', or with the independent evidence aggregation score γ^{indep} given by (4), symbolized by 'I'. It turns out that $\gamma^{\max}$ generally performs worse than the standard method of RAVE. The results of γ^{indep} are closer to those of the standard method of RAVE. Since γ^{indep} is not stable against duplication of snippets, these results profit from the fact that $\mathcal{I}_q^*$ does not contain exact duplicates (but there still can be overlapping snippets).

Table 5 shows the results of RAVE without active validation, i.e. when working with the original answer-support items in $\mathcal{I}_q$ only. Since RAVE aggregates results for answer keys rather than exact answer strings, it can exploit some minimal redundancy still present in the AVE08 test set. Therefore results are better than in the WF0 and WQ0 runs that use no aggregation at all, but considerably worse than those for support pool enhancement; see F-scores for RF (0.45), PRF (0.40) and WF0 (0.37).

Table 6 shows the results of RAVE when run with the prover switched off. There is a drastic loss in selection rate (e.g. a decrease by 15 percent points for SRQ compared to RQ), but surprisingly, a consistent improvement in the F-score. The SWF1 run even achieved the best F-score of all system variants of RAVE that were tested. These findings run counter to experience from many experiments on annotated CLEF07 data all of which show a better F-score when adding logic-based features [3, 5]. A possible

work. Synonym normalization of word senses was switched on for all questions, but should not be used for definition questions. The implementation of the important *awEatMatch* feature had to be corrected. Finally equation (8) was replaced by the improved (9).

Table 4: Results for alternative aggregation methods

model	f-score	f-gain	prec	p-gain	recall	qa-acc	qa-perf	sel-rate	s-gain
BRF	0.41	0.97	0.42	3.61	0.40	0.24	0.33	0.47	2.22
BWF.75	0.41	0.97	0.42	3.61	0.40	0.23	0.30	0.44	2.07
BWF1	0.41	0.97	0.42	3.61	0.40	0.24	0.33	0.47	2.22
IRF	0.45	1.16	0.42	3.63	0.48	0.26	0.34	0.50	2.37
IWF.75	0.47	1.26	0.44	3.83	0.50	0.26	0.34	0.50	2.37
IWF1	0.46	1.22	0.43	3.71	0.50	0.26	0.34	0.50	2.37
BRQ	0.39	0.89	0.33	2.86	0.48	0.32	0.32	0.61	2.91
BWQ.75	0.39	0.88	0.33	2.84	0.48	0.30	0.30	0.58	2.75
BWQ1	0.39	0.89	0.33	2.86	0.48	0.32	0.32	0.61	2.91
IRQ	0.43	1.06	0.35	3.01	0.55	0.33	0.33	0.63	2.98
IWQ.75	0.44	1.14	0.36	3.13	0.57	0.33	0.33	0.63	2.98
IWQ1	0.44	1.11	0.35	3.08	0.57	0.33	0.33	0.63	2.98

Table 5: Results without active validation

model	f-score	f-gain	prec	p-gain	recall	qa-acc	qa-perf	sel-rate	s-gain
PRF	0.40	0.94	0.42	3.69	0.38	0.22	0.29	0.42	1.99
PWF.75	0.41	0.98	0.43	3.74	0.39	0.22	0.29	0.42	1.99
PWF1	0.41	0.98	0.43	3.74	0.39	0.22	0.29	0.42	1.99
PRQ	0.38	0.84	0.32	2.81	0.46	0.29	0.29	0.56	2.68
PWQ.75	0.39	0.87	0.33	2.84	0.47	0.29	0.29	0.56	2.68
PWQ1	0.39	0.87	0.33	2.84	0.47	0.29	0.29	0.56	2.68

explanation is that eight of the eleven runs submitted for German were produced by QA systems which use variants of RAVE as the validator – and repeating the same, logic-based validation criterion already used when generating the runs is hardly effective in detecting the remaining false positives. The shallow-only approach, by contrast, is implemented by a different classifier trained on a larger data set.[12] It is probably only this independence benefit which reflects in the improved F-scores of the shallow technique.

Finally, Table 7 shows the results of the MAVE system [2] on the AVE 2008 test set, using clustering of answers ('C') and optimizing either F-score or qa-accuracy ('F' vs. 'Q'). Despite its use of a full logical answer-validation (i.e., hypothesis-snippet proofs), MAVE does not outperform RAVE with its simple logical passage test based on question-snippet proofs. The results may look disappointing compared to the 0.72 F-score and 0.93 selection rate of MAVE in the AVE 2007. However, the former F gain of 0.79 and selection gain of 1.8 in the AVE 2007 are considerably lower than current results in the AVE 2008.

Suitability for Real-Time Validation

Since RAVE is designed for real-time answer validation in interactive QA systems, the actual processing time needed for validation and aggregation is also of interest. For the standard method (RQ run), validation and aggregation/selection took an average 126 ms per question (or 9.35 ms per validation item).[13] When switching off the support pool enhancement (PRQ run), validation time drops to an average 76 ms per question (or 8.8 ms per validation item). For comparison, the validation time for shallow-only validation without using the prover is 77 ms per question (or 5.7 ms per validation item) in the SRQ run, and 50 ms

[12]The data set used for training the shallow-only classifier also includes all answer candidates supported by non-parseable snippets.

[13]These validation times do not include the time needed for parsing, since parsing reduces to the analysis of the question in normal operation of RAVE. For the same reason, the time needed for support pool enhancement was also excluded. All processing times were measured by running RAVE in a single thread on an Athlon64X2 4800+ CPU with 2.4 GHz clock rate.

Table 6: Shallow-only results

model	f-score	f-gain	prec	p-gain	recall	qa-acc	qa-perf	sel-rate	s-gain
SRF	0.50	1.42	0.42	3.67	0.61	0.23	0.30	0.44	2.07
SWF0	0.45	1.18	0.40	3.47	0.51	0.18	0.24	0.35	1.68
SWF.75	0.51	1.49	0.43	3.78	0.63	0.21	0.27	0.40	1.91
SWF1	0.52	1.50	0.43	3.76	0.64	0.23	0.30	0.44	2.07
SRQ	0.46	1.21	0.35	3.05	0.65	0.26	0.26	0.50	2.37
SWQ0	0.41	0.99	0.32	2.82	0.56	0.23	0.23	0.44	2.07
SWQ.75	0.47	1.27	0.36	3.14	0.67	0.24	0.24	0.47	2.22
SWQ1	0.47	1.28	0.36	3.13	0.68	0.26	0.26	0.50	2.37

Table 7: Results of MAVE on the AVE08 test set

model	f-score	f-gain	prec	p-gain	recall	qa-acc	qa-perf	sel-rate	s-gain
CF	0.42	1.05	0.32	2.80	0.61	0.27	0.30	0.52	2.45
CQ	0.41	1.00	0.31	2.67	0.63	0.29	0.29	0.55	2.60

per question (or 5.8 ms per validation item) without support pool enhancement. The extra effort required for applying the prover to a validation item suitable for logical processing was an average 5.4 ms.

4 Conclusion

The main objective for the current work was developing a logic-based answer validator suitable for use in real-time QA. The design of RAVE results from the premise that optimizing the prover is not sufficient to achieve this goal. Therefore the RTE paradigm for answer validation was replaced by a simplified model which only uses logic for passage validation. The actual processing times of RAVE on the AVE 2008 data confirm that the method can be applied in real-time QA even for large numbers of answer candidates.

Another key feature of RAVE is the replication-tolerant aggregation model. Redundancy is usually helpful for answer validation. However, redundancy only means stronger evidence when the sources are sufficiently independent. In particular, multiple copies or variants of the same document (e.g. news report or press release) should not provide more weight than a single occurrence of the original document. The aggregation model of RAVE leverages redundancy without being misled by replicated content. Experiments confirm the superiority of the model over several alternative approaches.

An interesting aspect of RAVE is the discernment of regular and auxiliary support passages; the latter are only used for aggregation and never actually shown to the user. For example, snippets from corpora with restrictive licensing can be valuable sources of information but may not be presented to the user for legal reasons. The same holds for supporting passages in a language unknown to the user; in this case, presenting the snippet in the final result would be pointless.

RAVE achieved an F-score of 0.39 and qa-accuracy of 0.32 in the AVE 2008. These scores are lower than those of MAVE in the last year. However, the test sets for German had very different characteristics in these two years, and the F-score and qa-accuracy are not commensurable for different test sets. The F-score gain compared to the 100% YES baseline [9], and the selection gain compared to random selection, are better suited for a comparison across test sets since they relate the validation and selection results to the obvious baselines. From this perspective, the RAVE results for 2008 even look better than those of MAVE for 2007. Moreover RAVE outperforms MAVE on the 2008 test set. It must be admitted, though, that the best individual QA system in the test set performed better than RAVE: It achieved a qa-accuracy of 0.38 (or selection rate of 0.73), compared to the qa-accuracy of 0.34 and selection rate of 0.65 of RAVE in the RQ run. This clearly indicates that improvements of the validator are necessary. Generally speaking, RAVE has good features for recognizing passages with a correct answer but (currently) poor means for verifying that a given candidate equals this correct answer. Therefore the most important change to RAVE will be the addition of features which judge the compatibility of the answer candidate with the result of the question-passage proof. The answer-type check must also be improved since the current implementation is very limited.

On the other hand, the AVE imposed some adverse conditions on the validator so that RAVE will likely perform better in actual applicatons. Firstly, the features used by RAVE normally include the retrieval score of the passage retrieval system and a producer score assigned by the QA source; these features are not available in the AVE test set. Cross-validation on the training set shows that support for these features might improve the F score by up to 8 percent points. Second, the AVE development set for German was too small for training the classifiers. Therefore an existing training set of sufficient size had to be chosen whose characteristics are not necessarily similar to those of the AVE 2008 test set. In practice, RAVE is normally adjusted to each QA source by learning a separate model for each QA system. Such customization was not possible in the AVE since the QA sources and their characteristics were not disclosed.

RAVE is already part of two question answering systems. It serves as the answer validator in a multi-stream system (IRSAW) [7], where it provides incremental, any-time answer validation capability. Moreover RAVE is the core of the evolving LogAnswer system [1, 5], which uses the question-passage proofs of RAVE for simultaneously extracting answer bindings and validating the corresponding answers. It is too early at this time to decide which of the two approaches is superior to the other. However, the existing integration of RAVE in both kinds of systems will make it possible to address this issue in the near future.

References

[1] Ulrich Furbach, Ingo Glöckner, Hermann Helbig, and Björn Pelzer. The LogAnswer project at QA@CLEF 2008. In *Working notes for the CLEF 2008 workshop*, 2008.

[2] Ingo Glöckner. University of Hagen at QA@CLEF 2007: Answer validation exercise. In *Working Notes for the CLEF 2007 Workshop*, Budapest, 2007.

[3] Ingo Glöckner. Towards logic-based question answering under time constraints. In *Proc. of the 2008 IAENG Int. Conf. on Artificial Intelligence and Applications (ICAIA-08)*, Hong Kong, 2008.

[4] Ingo Glöckner, Sven Hartrumpf, and Johannes Leveling. Logical validation, answer merging and witness selection: A study in multi-stream question answering. In *Proc. RIAO-07*, Pittsburgh, 2007.

[5] Ingo Glöckner and Björn Pelzer. Exploring robustness enhancements for logic-based passage filtering. In *Proc. of KES-2008*, Lecture Notes in Computer Science. Springer, 2008 (to appear).

[6] Sven Hartrumpf. *Hybrid Disambiguation in Natural Language Analysis*. Der Andere Verlag, Osnabrück, Germany, 2003.

[7] Sven Hartrumpf, Ingo Glöckner, and Johannes Leveling. University of Hagen at QA@CLEF 2008: Efficient Question Answering with Question Decomposition and Multiple Answer Streams. In *Working notes for the CLEF 2008 workshop*, 2008.

[8] Hermann Helbig. *Knowledge Representation and the Semantics of Natural Language*. Springer, 2006.

[9] Anselmo Peñas, Álvaro Rodrigo, Valentin Sama, and Felisa Verdejo. Testing the reasoning for question answering validation. *Journal of Logic and Computation, Special Issue on Natural Language and Knowledge Representation*, to appear.

[10] Ian H. Witten and Eibe Frank. *Data Mining. Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.