Multi-dimensional feature merger for Question Answering

Apoorv Agarwal[1] *J. William Murdock*[2]
Jennifer Chu-Carroll[2] *Adam Lally*[2] *Aditya Kalyanpur*[2]
(1) Department of Computer Science, Columbia University, New York, NY, U.S.A.
(2) IBM T.J. Watson Research Center, Yorktown Heights, NY, U.S.A.
apoorv@cs.columbia.edu, {murdockj, jencc, alally,
adityakal}@us.ibm.com

Abstract

In this paper, we introduce new features for question-answering systems. These features are inspired by the fact that justification of the correct answer (out of many candidate answers) may be present in multiple passages. Our features attempt to combine evidence from multiple passages retrieved for a candidate answer. We present results on two data-sets: Jeopardy! and Doctor's Dilemma. In both data-sets, our features are ranked highest in correlation with gold class (in the training data) and significantly improve the performance of our existing QA system, Watson.

Keywords: Question Answering, multi-dimensional feature merger, Watson.

1 Introduction

Most existing factoid question answering systems adopt search strategies and scoring algorithms with the assumption that a short passage exists in the reference corpus which contains sufficient information to answer each question. This assumption largely holds true for short and focused factoid questions such as those found in the TREC QA track (Voorhees and Tice, 2000). Examples of TREC QA questions include "*When did Hawaii become a state?*" and "*What strait separates North America from Asia?*" However, some more complex factoid questions contain facts encompassing multiple facets of the answer, which often cannot be found together in a short text passage. Consider the following examples, selected from collections of Jeopardy![1] and Doctor's Dilemma[2] questions, respectively:

(1) WHO'S WHO IN SPORTS: Born in 1956, this Swedish tennis player won 6 French Opens & 5 straight Wimbledons (A: Björn Borg)

(2) CARDIOLOGY: Murmur associated with this condition is harsh, systolic, diamond-shaped, and increases in intensity with Valsalva (A: Hypertrophic cardiomyopathy)

In both examples, information presented in the question can reasonably be expected to be in documents that describe the respective answer entities. However, it is quite unlikely that all the information will be present in one or two adjacent sentences in the document. More specifically, in example (1), we find birth year and nationality information in the basic biographic section of documents about Björn Borg, while statistics about his tennis record can generally be found in a section about Borg's career. Similarly, for example (2), the descriptions of typical murmurs associated with hypertrophic cardiomyopathy (harsh, systolic, and diamond-shaped) may not fall under the same section as the impact of Valsalva maneuver on the murmur (which is a factor used to distinguish hypertrophic cardiomyopathy from aortic stenosis). As a result, a typical passage retrieved from most reference corpus would cover only a portion of the facts given in the question.

These multi-faceted factoid questions present a challenge for existing question answering systems which make the aforementioned assumption. Consider the following short passages relevant to the question in example (2):

(2.1 a) Hypertrophic cardiomyopathy generates a harsh late-systolic murmur, ending at S2.

(2.1 b) The straining phase of the Valsalva maneuver induces an increase in the intensity of the systolic ejection murmur of hypertrophic cardiomyopathy.

(2.2 a) A harsh, late-peaking, basal murmur radiating to the carotid arteries suggests aortic stenosis.

(2.2 b) A classic physical finding of aortic stenosis is a harsh, crescendo-decrescendo systolic murmur that is loudest over the second right intercostal space and radiates to the carotid arteries.

Existing systems which evaluate each passage separately against the question would view each passage as having a similar degree of support for either hypertrophic cardiomyopathy or aortic

[1] http://www.jeopardy.com; Jeopardy! is a registered trademark of Jeopardy! Productions, Inc.

[2] http://www.acponline.org/residents_fellows/competitions/doctors_dilemma

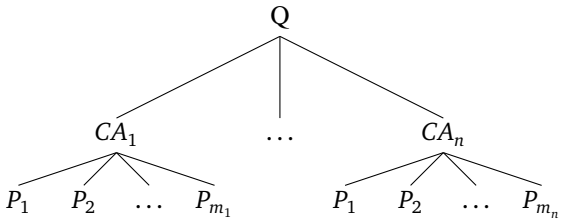

Figure 1: Typical question answering scenario. *Q* refers to question. *CA* are candidate answers for question *Q*, and *p* refers to passages supporting candidate answers.

stenosis as the answer to the question. However, these systems lose sight of a crucial fact, namely, that even though each passage covers half of the facts in the question, (2.1 a) and (2.1 b) cover disjoint subsets of the facts, while (2.2 a) and (2.2 b) address the same set of facts.

In this paper, we introduce the notion of *multi-dimensional feature merger* or *MDM* features, which allow for passage scoring results to be combined across different dimensions, such as question segments and different passage scoring algorithms. In this motivating example, MDM features that combine results across question segments would capture the broader coverage of passages (2.1 a) and (2.2 b), and thus enable the system to recognize hypertophic cardiomyopathy as a better answer for the question than aortic stenosis. We describe a general-purpose MDM feature merging framework that can be adopted in question answering systems that evaluate candidate answers by matching candidate-bearing passages against the question. We discuss our implementation of this MDM feature merging framework on top of our own question answering system, Watson. Finally, we demonstrate how passage scoring results can be merged across various dimensions in our system, resulting in 1) new features that are more highly correlated with correct answers than the base features from which they were derived, and 2) significant component level performance improvement and 3) end-to-end performance improvement. We present a comprehensive set of experiments for our current domain of interest – the medical domain and a less comprehensive set of experiments for Jeopardy! data.

The rest of the paper is organized as follows. In section 2, we describe our feature set. Since we build on existing state-of-the-art QA system, in section 3, we briefly describe the current system, focusing on the component of the system that we enhance in this paper. In section 4, we describe passage scorers in the current system, with specific examples of features that leverage scores assigned to passages by these scorers. In section 5, we presents a detailed description of the data we use for training and testing. Additionally, we present experiments and results to show the impact of our features. Section 6 presents a survey of current work in question answering. Finally, we conclude and present future direction of research in the last section.

2 Multi-dimensional feature merger (MDM)

Given a question, *Q*, each of its candidate answer, *CA*, has a set of supporting passages (Figure 1). In a typical question-answering system, support of each passage for a candidate answer is quantified. Then a merging strategy is used to combine the support of all passages for a particular candidate answer. In this paper, we introduce a general framework for merging support from supporting passages.

The methodology of calculating the support of a passage for a candidate answer is called *passage scoring* (Murdock et al., 2012a). At an abstract level, a passage scorer is responsible for quantifying how well a passage *matches* a question. We represent a question and a passage as an ordered set of

sum($\vec{s}$)	avg($\vec{s}$)	std($\vec{s}$)	max($\vec{s}$)	min($\vec{s}$)	non-zero($\vec{s}$)
$\sum_{j=1}^{cols} s_j$	$\frac{sum(\vec{s})}{cols}$	$\sqrt{\frac{\sum_{j=1}^{cols}(s_j - avg(\vec{s}))^2}{cols-1}}$	$\arg\max_{j \in [1,cols]} s_j$	$\arg\min_{j \in [1,cols]} s_j$	$\lvert\{s_j \lvert s_j \neq 0 \forall j \in [1, cols]\}\rvert$

Table 1: Standard formulae that constitute $g(M)$

Question	large	land	animal	has	large	ears	
P1.1	x_1	x_2	x_3	x_4	x_5	x_6	$\vec{f_{1.1}}$
P1.2	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	$\vec{f_{1.2}}$
P2.1	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	$\vec{f_{2.1}}$
P2.2	x_{19}	x_{20}	x_{21}	x_{22}	x_{23}	x_{24}	$\vec{f_{2.2}}$

Table 2: Passage match scores for question and passages in Figure 2.

terms ($Q = \{q_1, q_2, \ldots, q_n\}$), and ($P = \{p_1, p_2, \ldots, p_m\}$) respectively, Passage scorers align question terms to passage terms and assign a *score* based on how well the terms align. For example, a passage scorer will take as input Q and P and output a vector of scores that represents how well the passage matches the question. We denote this vector for P as $\vec{f}$ such that f_i is the score of how well one of the passage terms matches the i^{th} term in the question. Note the length of this vector is fixed per question but may vary across questions.

We collect all these vectors per question, per candidate answer into a matrix, M. For example, CA_1 may be represented as a matrix where row i corresponds to the passage scoring vector for passage P_i. An element of this matrix, $f_{i,j}$ is the score assigned by one of the passage scorers of how well passage P_i aligns with the term j in the question Q.

This matrix is of variable dimensions for different candidate answers per question. Number of rows could be different because the number of supporting passages could be different for each candidate answer for the same question. Since different questions have different number of question terms, the number of columns could be different for candidate answers across questions. Therefore, we cannot capture the distribution of this matrix simply by linearizing the matrix.

In this paper, we define a function $f : M \rightarrow \mathbf{R}^N$, that maps each matrix into feature vector of fixed length, N. This function is defined as follows:

$$f(M) = < g(M), g(M') >$$

where M' is the transpose of matrix M and g is a function $g : M \rightarrow \mathbf{R}^{N/2}$ that maps a matrix into feature vector of fixed length, defined as follows:

$$g(M) = <\text{sum}(\vec{s}), \text{avg}(\vec{s}), \text{std}(\vec{s}), \text{max}(\vec{s}), \text{min}(\vec{s}), \text{dim}(\vec{s}), \text{non-zero}(\vec{s})>$$

where $\vec{s}$ is a vector of dimensionality dim($\vec{s}$), such that $s_j = \sum_{i=1}^{rows} f_{i,j}$ and the remaining standard formulae are given in Table 1.

Consider an example Jeopardy! question:[3] *This large land animal also has large ears.* Consider two candidate answers and their supporting passages:

[3] modified for readability.

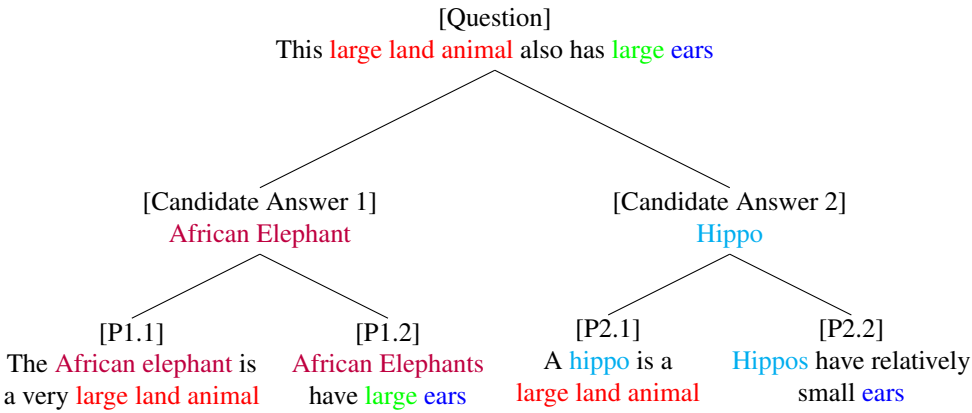

Figure 2: A specific example showing candidate answers and supporting passages for a modified Jeopardy! question. P1.1 means first justifying passage for the first candidate answer.

1. Candidate answer 1: African Elephant
   (a) P1.1: The African Elephant is a very large land animal.
   (b) P1.2: African elephants have large ears.
2. Candidate answer 2: Hippo
   (a) P2.1: A hippo is a large land animal.
   (b) P2.2: Hippos have relatively small ears.

This example is shown pictorially in Figure 2

Table 2 abstractly shows how passage scorers assign values to specific question terms for specific passages. For example, consider the P1.1 row, which represents how well the passage *The African elephant is a very large land animal* supports the answer *elephant* for the question *This large land animal also has large ears*. If the passage scorer is effective, it will give a high score to x_1, x_2 and x_3 (because the passage does, indeed, provide strong justification for "elephant" satisfying the requirements of being large land animal). It will give a very small score (typically 0) to x_4, x_5, and x_6, because the passage says nothing about elephants having large ears. However, some passage scorers may be mislead by the fact that the term "large" appears twice question and either one could align to the one occurrence in the passage. Often some passage scorers match too many terms and thus assign credit to terms that don't deserve it while others match too few and miss important content; this is why we have a diverse collection of scorers and let the classifier sort out how much to trust each of them.

Using one of the existing merging strategy, say MAX, candidate answer 1, African Elephant, will get assigned a feature value equal to $MAX\{(x_1+x_2+x_3+x_4+x_5+x_6),(x_7+x_8+x_9+x_{10}+x_{11}+x_{12})\}$. So either passage P1.1 or passage P1.2 will be selected as an optimal passage. As is apparent from this merger strategy, it does not attempt to leverage the complementary information in the two passages. Our merging strategy will attempt to capture the distribution of alignment across passages. For the matrix for African Elephant, M, $f(M) =< g(M), g(M') >$. First dimension of vectors

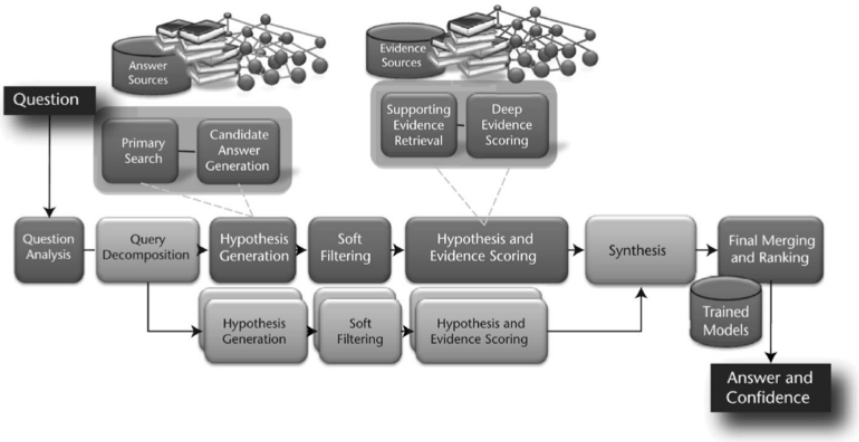

Figure 3: Architecture of Watson, state-of-the-art DeepQA system (taken from (Ferrucci et al., 2010)).

$g(M)$ and $g(M')$ will be the same, because $sum(\vec{s}) = sum(\vec{s}') = \sum_{i=1}^{12} x_i$. But others will be different. For example, $mean(\vec{s}) = \frac{1}{6} * sum(\vec{s})$, whereas, $mean(\vec{s}') = \frac{1}{2} * sum(\vec{s})$.

Note, the $sum(\vec{s})$ feature is aggregating the information across passages. In a passage scorer, which assigns 1 for a match and 0 otherwise, it is clear why this feature will have a higher value for African Elephant, the correct answer, than Hippo (because Hippo's don't have *large* ears).

Our framework is general in three ways: 1) It is independent on the type of passage scorer, 2) More matrix operations (like *rank(M)*), may be easily added to the definition of function $g(M)$, and 3) Our framework is easily extensible to beyond two dimensions, which can be used to capture additional orthogonal feature dimensions (see future work section for an example).

In the following sections, we first describe a specific, and state-of-the-art QA system, Watson. We present where our features fit in the larger architecture. Then we give an overview of specific passage scorers and merging strategies in the current system, followed by experiments and results showing that the new features we introduce add value to the current system.

3 Overview of Watson

IBM undertook the challenge to build a question-answering system named *Watson* that is able to answer open domain questions, such as those posed in a U.S. quiz show Jeopardy!. An overview of the architecture of Watson is illustrated in Figure 3. We refer the reader to (Ferrucci et al., 2010) for a detailed description of the architecture. In this section, we present a high level overview of the system pointing out where our features fit in.

The DeepQA system analyzes a question, *Question Analysis* (Lally et al., 2012), and generates multiple possible candidate answers, *Hypothesis Generation* (Chu-Carroll et al., 2012).It then applies many different answer scoring algorithms, each of which produces features that are used to evaluate whether the answer is correct. One way in which DeepQA evaluates candidate answers is to first retrieve passages of text that contain the candidate answer, via a technique called Supporting Evidence Retrieval; each passage is then scored using a variety of algorithms called passage scorers

$$< Q_1, CA_1, -1 >, < Q_1, CA_2, -1 >, \ldots, < Q_1, CA_i, 1 >, \ldots, < Q_1, CA_{n_1}, -1 >$$
$$< Q_2, CA_1, -1 >, < Q_2, CA_2, -1 >, \ldots, < Q_2, CA_j, 1 >, \ldots, < Q_2, CA_{n_2}, -1 >$$
$$\ldots$$
$$< Q_m, CA_1, -1 >, < Q_m, CA_2, -1 >, \ldots, < Q_m, CA_k, 1 >, \ldots, < Q_m, CA_{n_m}, -1 >$$

Figure 4: Training and test data for a question-answering system. Each question Q has multiple candidate answers, CA, where few, if any, are correct (class = 1).

in the *Hypothesis and Evidence Scoring* phase (Murdock et al., 2012a). All of the features are sent to a *Final Merging and Ranking* (Gondek et al., 2012) component, which uses machine learning techniques to weigh and combine features to produce a single confidence value estimating the probability that the candidate answer is correct. The features we introduce are extracted and made available to the machine learning model in the Final Merging and Ranking component, where the scores assigned by different passage scorers are available. In the next section 4, we give details of existing passage scorers and their feature merging strategies used prior to the framework introduced in this paper.

4 Passage scoring

Our question-answering system works by finding candidate answers, employing a variety of algorithms to compute feature values relating to those answers, and then using a statistical classifier to determine which candidate answer is correct. A question-answering scenario is shown in Figure 1. For a given question Q, search components find a set of candidate answers $\{CA_1, CA_2, \ldots, CA_n\}$. The task of the classifier is to decide which of the candidate answers is the correct answer. Hence the training and test data for that classifier looks as in Figure 4.

Each candidate answer is associated with one or more passages that contain the candidate answer. A subset of the algorithms that compute feature values in our system are the passage scoring components. These components evaluate the evidence that a single passage provides relating to how well the candidate answer satisfies the requirements of the question. Thus among the feature values associated with a candidate answer, some will be passage scoring features.

Our passage scorers are described in detail elsewhere (Murdock et al., 2012a). Here we provide only a brief introduction to provide context for later sections of this paper. We have a variety of passage scoring algorithms that use different strategies for determining which parts of a question to attempt to match to each part of a passage and for determining whether two parts of a passage match. Some attempt to align question terms to passage terms using syntactic structure and/or semantic relations, while others use word order or ignore the relationship among terms completely (e.g., simply counting how many question terms appear in the passage, regardless of whether those terms are similarly arranged).

Watson's passage scorers leverage available annotation components developed for the DeepQA framework, such as dependency parsing, Named Entity (NE) recognition, coreference resolution and relation detection. The question and the passage are decomposed into sets of terms, where a term can either be a single token or a multiword token. All of these scorers try to determine the amount of overlap between the passage and the question by looking at which terms match. The individual scorers put different restrictions on when a term is considered to *match*.

Currently, there are four scorers being used in the system:

1. **Passage Term Match**: Assigns a score based on which question terms are included in the passage, regardless of word order or grammatical relationship.

2. **Skip Bigram**: Assigns a score based on whether pairs of terms that are connected or nearly connected in the syntactic-semantic structure of the question match corresponding pairs of terms in the passage.

3. **Textual Alignment**: Assigns a score based on how well the word order of the passage aligns with that of the question, when the focus is replaced with the candidate answer.

4. **Logical Form Answer Candidate Scorer** (LFACS): Targets high-precision matching between the syntactic structures of passages and questions, and is therefore quite restrictive concerning structural overlap of the question and the passage. Like Skip Bigram, it operates on syntactic-semantic structural graphs, which contain one node for each lexical item.

Each passage scoring component produces a fixed number of feature value pairs for each candidate answer within each passage. Some of these values range from 0 to 1, where a high score indicates that the passage matches the question well based on that passage scorer's evaluation criteria; other passage scorers have other ranges. Watson's final answer merging and ranking component considers a pre-defined set of features and applies a machine learned model to score each candidate answer. However, since each candidate has multiple, and generally a varying number of supporting passages, we use a merger to combine passage scores for $<$ candidate answer, passage $>$ pairs into a fixed set of features. For example, if a candidate answer has three passages and a passage scorer assigns a value of 0.5, 0.6, and 0.7 to each passage, these scores may be merged using a merger strategy like MAX. Using this merger strategy, the feature added to the learning model for the candidate answer under consideration will be $MAX(0.5, 0.6, 0.7) = 0.7$.

We have the following three distinct algorithms that we use to merge features across passages (Gondek et al., 2012).

1. Maximum: The final score for the candidate answer is the maximum score for that answer in any passages found for that answer.

2. Sum: The final score for the candidate answer is the sum of the scores for that answer in each of the passages found for that answer.

3. Decaying sum: The final score for the candidate answer is computed to be $\sum_{i=0}^{m} \frac{p_i}{2^i}$, where $p_0, p_1, \ldots, p_m$ are the scores of the passages that contain the answers, sorted in descending order.

A key limitation of our earlier work is that the passage scorers capture limited complementary information that the passages have to offer. For example, in Figure 2, a passage scoring component may assign scores $s_{1.1}, s_{1.2}$ to passages P1.1 and P1.2 respectively. A merger strategy that takes maximum across passages will choose $MAX(s_{1.1}, s_{1.2})$ as the optimal supporting passage. However, since these passages have complementary information to offer, it would be better to somehow aggregate this information. This is exactly where our multi-dimensional merging features come into the picture.

As described in earlier publications (Gondek et al., 2012), for each of our features, we have two other derived features: a feature for whether that feature is missing and a standardized version of the

Feature name	Explanation	In terms of Table 2
MDM-TextualAlignment-sum-then-mean	For each question term, compute the sum of the Textual Alignment scores across all passages, and then compute the mean of the sums	$f(M) = [(x_1 + x_7) + (x_2 + x_8) + (x_3 + x_9) + (x_4 + x_{10}) + (x_5 + x_{11}) + (x_6 + x_12)]/6$
MDM-SkipBigram-transpose-sum-then-mean	For each passage, compute the sum of the Skip-Bigram scores across all question terms, and then compute the mean of the sums	$f(M) = [(x_1 + x_2 + \ldots + x_6) + (x_7 + x_8 + \ldots + x_{12})]/2$
MDM-LFACS-max-then-sum	For each question term, compute the maximum of the LFACS scores across all passages, and then compute the mean of the maxima	$f(M) = max(x_1, x_2, \ldots, x_6) + max(x_7, x_8, \ldots, x_{12})$
MDM-SkipBigramScore-transpose-sum-then-nonZeroColumns	For each passage, compute the sum of the Skip-Bigram scores across all question terms, and then compute the number of sums that are non-zero	Set $cnt = 0$. If $(x_1 + x_2 + \ldots + x_6) > 0, cnt = cnt + 1$. If $(x_7 + x_8 + \ldots + x_{12}) > 0, cnt = cnt + 1$. $F(M) = cnt$.

Table 3: Examples of MDM features. First column is the feature name, column 2 a natural language description of the feature and the third column is the exact mathematical formula in reference to Table 2 for passages P1.1 and P1.2 belonging to the candidate answer 1.

feature. When the value of a feature is missing, we assert a value of 0 for the feature and a value of 1 for the corresponding derived missing feature; this allows the learner to distinguish between cases where the feature actually has 0 value versus cases where it simply did not apply at all. The standardized version of a feature is computed by subtracting the mean value of that feature and dividing by the standard deviation for that feature. Both mean and standard deviation are computed across all answers to a single question, *not* across all answers to all questions in the test set. The purpose of the standardized feature is to encode how much the base feature differs from a typical value of that feature for a single question.

In Table 3, we present examples of some top scoring (in terms of correlation with the gold class) MDM features. For a passage scoring feature X, we produce the following MDM features: MDM-X-sum-then-mean ($avg(\vec{s})$), MDM-X-transpose-sum-then-mean ($avg(\vec{s'})$), MDM-X-sum-then-max ($max(\vec{s})$) etc.

5 Experiments and Results

To demonstrate the generality of our approach, we experimented with two data sets, an open-domain question set and one focused on the medical domain. We briefly describe these data sets in this section. Our first open-domain test set is a randomly selected set of 3,505 Jeopardy! questions. Jeopardy! questions span a large number of domains, including arts and entertainment, history, geography, and science. These questions are also generally more complex, incorporating multiple loosely related facts about the correct answers, particularly as compared with typical questions from the TREC QA track. The last characteristic makes Jeopardy! questions an excellent test set for our MDM feature merging framework.

Our second test set is a collection of 905 Doctor's Dilemma questions. Doctor's Dilemma, also

	#Questions	#Positive	#Negative	#Average cand. per Q
Jeopardy!	11,520	12,173	2,555,396	222.87
Doctor's Dilemma	1,322	2,338	543,963	413.23

Table 4: Data distribution for our data-sets. #Question refers to number of questions. #Positive refers to number of positive instances i.e. correct answers to questions, #Negative refers to number of negative instances and #Average cand. per Q refers to the average number of candidates considered for a particular question. Note, this is simply total number of positive and negative examples divided by the number of questions in the data-set.

known as Medical Jeopardy, is a competition organized by the American College of Physicians for medical interns and residents and held each year at the Internal Medicine meeting. The format of these questions is modeled after Jeopardy!, while their content is focused solely on topics related to medicine. Although not as linguistically complex as Jeopardy! questions, Doctor's Dilemma questions generally also consists of multiple facts about the correct answer, making it suitable as a test set for MDM features. Following are some examples from the Doctor's Dilemma domain:

1. The syndrome characterized by joint pain, abdominal pain, palpable purpura, and a nephritic sediment. Answer: Henoch-Schonlein Purpura.

2. Familial adenomatous polyposis is caused by mutations of this gene. Answer: APC Gene.

3. The syndrome characterized by narrowing of the extra-hepatic bile duct from mechanical compression by a gallstone impacted in the cystic duct. Answer: Mirizzi's Syndrome.

We use a supervised learning paradigm, with features extracted as described in previous sections. We use logistic regression classifier for training and testing. We report results on a held-out test set for both data-sets. The distribution of training set for the two data-sets are in Table 4. We test on 3,505 Jeopardy! questions and 905 DD questions.

We present three types of analyses to show the usefulness of our features. First, we present the correlation of our features with the gold class (for the training set only) i.e. correctness of a candidate answer. Second, we present a *component level analysis*, where we add our features to a baseline QA system and show improvement. Third, we present results on the end-to-end Watson system.

5.1 Correlation

A standard way to judge the goodness of features is to look at the features' Pearson's r correlation with the gold class (Hall, 2000). The Pearson's r correlation coefficient between feature X and gold standard Y is given by:

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

where $\bar{X}$ and $\bar{Y}$ are the arithmetic mean of feature values and gold class values respectively. We refer to the degree of correlation between the feature and the gold class as the "informativeness" of the feature. Naturally, we would like to keep features that have high informativeness.

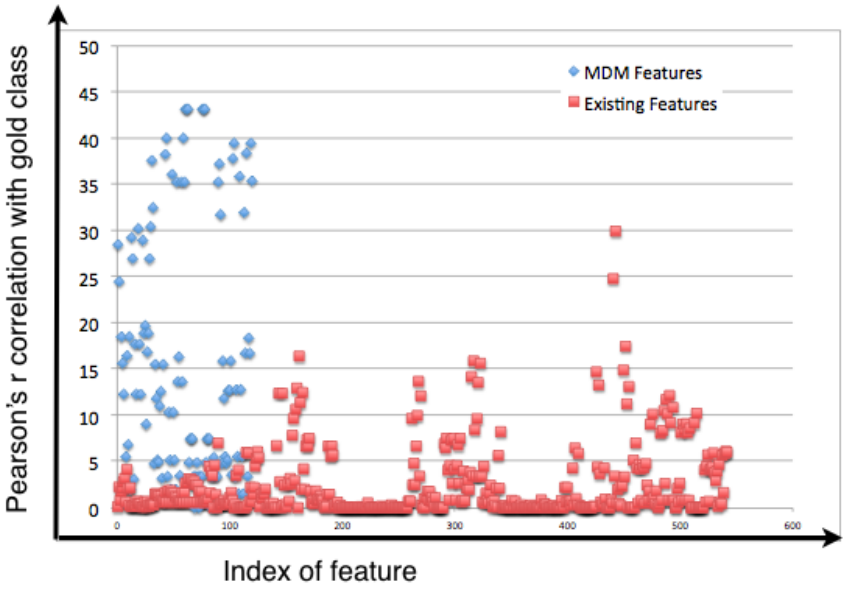

Figure 5: Inform analysis comparison of MDM features with the existing features in the system trained on **Jeopardy!** data. X-axis is the feature index (in no specific order) and Y-axis is the % correlation of features with the gold class.

Figure 5 presents the informativeness of existing features (red squared dots) and MDM features (blue diamond dots) for the Jeopardy! data-set. In figure 5, the x-axis is the feature index (existing features indexed from 1 to 535 and new features indexed from 1 to 110) and the y-axis is the informativeness of the features. For example, the highest informativeness of existing features (square red dot) is 30% ($100 \cdot r$), while the highest informativeness of MDM features is 43.2%. Many of the MDM features have higher informativeness than the most correlated feature in the existing system.

Similar is the case with the medical domain data. Figure 6 presents the informativeness of existing features (red squared dots) and MDM features (blue diamond dots) for the Doctor's Dilemma data-set. The highest informativeness of MDM features is 21.5%, which is comparable to the three existing features with highest informativeness (between 20% to 21%). However, as the graph shows, the vast majority of MDM features have substantially higher informativeness than the original features. the Jeopardy! domain, many of the MDM features are more correlated with answer correctness than most of the original features.

5.2 Component level analysis

As described in section 3, we add new features in the final merger stage of the system. Our features are calculated for each of the four passage scorers described in section 4. In this section, we evaluate the impact of these MDM features when only a single passage scoring component is employed in the system. To do so, we create a *component level baseline* for each of our four passage scorers as follows: on top of the Watson answer-scoring baseline configuration(Ferrucci et al., 2010), which includes all of the standard question analysis, search, and candidate generation, but only one answer scorer (which checks answer types using a named entity detector (Murdock et al., 2012b)) and a simplified configuration for merging and ranking answers. We add each of our existing Passage

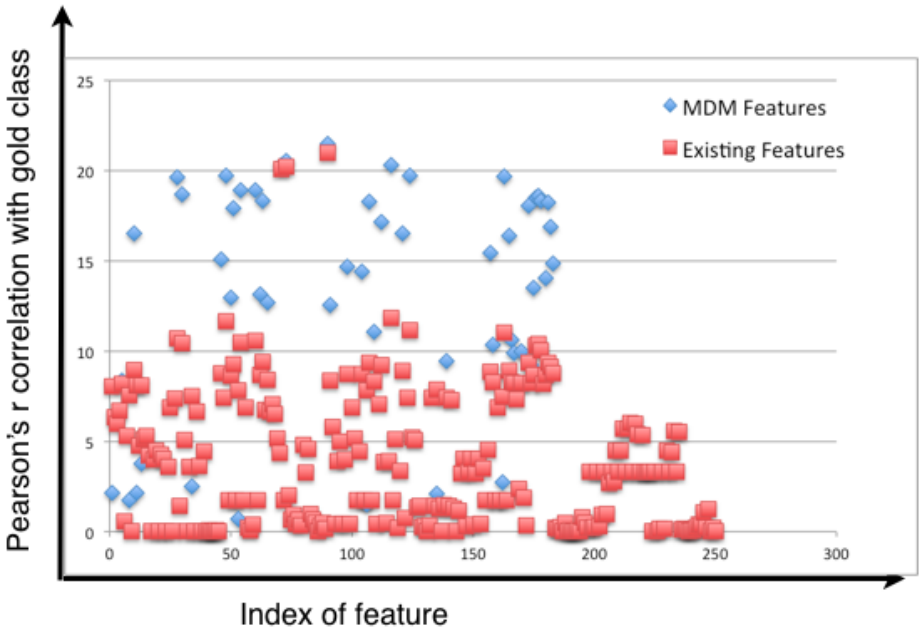

Figure 6: Inform analysis comparison of MDM features with the existing features in the system trained on **Doctor's Dilemma** data. X-axis is the feature index (in no specific order) and Y-axis is the % correlation of features with the gold class.

Term Match, Skip Bigram, Textual Alignment, and LFACS passage scoring, to create four baseline systems. We then compare each baseline to the system with our MDM features for the corresponding passage scorer and show a significant gain in Precision@70% and accuracy.

We often consider Precision@70% as a numerical measure that combines the ability to correctly answer questions and the ability to measure confidence; this metric corresponds to the precision when the system answers 70% of the questions of which it is most confident.

Table 5 present results for our component level analysis for Doctor's Dilemma questions. A component level baseline for each passage scorer was computed as described above. System performance improves across the board after adding MDM features for a passage scorer. Using

	Component Level Baseline		With MDM features	
Passage Scorer	Precision@70%	%Accuracy	Precision@70%	%Accuracy
Passage Term Match	24.9	20.2	29.2	23.4
Skip Bigram	26.8	21.5	28.7	23.3
Textual Alignment	22.9	18.8	25.7	21.1
LFACS	25.7	20.3	28.5	22.4

Table 5: Component level comparison for Doctor's Dilemma data-set for each of the four passage scorers. Each component level baseline is the answer-scoring baseline plus features for one of the passage scorers. All the numbers after adding MDM features for a passage scorer are significantly better than the baseline by $p < 0.05$, using McNemar's significance testing.

	Baseline		With MDM features	
Data-set	Precision@70%	%Accuracy	Precision@70%	%Accuracy
Doctor's Dilemma	37.2	29.2	40.2	31.3

Table 6: End-to-End comparison for medical domain data, Doctor's Dilemma. Baseline refers to the configuration with all the current features in the system. With MDM features refers to the configuration when we add all our MDM features to the existing feature set. This difference in performance is statistically significant with $p < 0.05$, using McNemar's significance testing.

McNemar's significance test, these are statistically significant improvements over the baseline at $p < 0.05$. As is clear from the results, for each of the four passage scorers, adding MDM features that capture the distribution of the passage scores across multiple passages improves the performance, in terms of both Precision@70% and % accuracy, by a significant amount.

For the Jeopardy! data-set, for the LFACS passage scorer, Precision@70% improves from 64.9% to 71.3% and % Accuracy improves from 52.2% to 57.3%. Both these improvements are statistically significant at $p < 0.05$, using McNemar's significance testing.

Based on these experimental results, we conclude that addition of MDM features for passage scorers significantly improves the performance of our QA system.

5.3 End-to-End Analysis

In this section, we present results for running the full Watson system with and without MDM features. Table 6 shows the Precision@70% and % accuracy performance on the Doctor's Dilemma test set. The results show that by adding MDM features to existing system, we are able to get a statistically significantly better performance than the baseline system: Precision@70% improves from 37.2 to 40.2 and % accuracy improves from 29.2% to 31.3%.

6 Literature Survey

Question answering has had a long history (Simmons, 1970) and has seen considerable advancement over the past decade (Maybury, 2004; Strzalkowski and Harabagiu, 2006). However, to the best of our knowledge, there is no general purpose framework integrated into a QA system that is capable of aggregating information across multiple pieces of evidence, each analyzed using different analytics (features), and comparing this with coverage of terms/facts in the input question.

A technique that is complementary to ours is corpus expansion (Schlaefer et al., 2011), in which corpus documents are expanded to include topically related facts from an external resource (e.g. Web). Sometimes in this process, pseudo documents are created which contain aggregate information about a particular entity. This approach helps standard document search by providing better document-level evidence/scores for the input search terms. The system is more likely to find a single document that addresses all of the parts of the question in a corpus after it has been expanded. However, passage scoring still encounters the same underlying problem even with an expanded corpus: in some cases, there will not be any single passage that addresses all of the requirements of the question.

The second related approach is question decomposition (Kalyanpur et al., 2012; Felshin, 2005), which aims at decomposing the question into different facts that need to be independently or sequentially solved in order to arrive at the correct answer. However, question decomposition does not deal with the issue of combining multiple pieces of evidence (possibly assessed using different

analytics) for the same fact within a decomposed question (which our approach does). In addition, the process of decomposing a question into multiple subquestions is an extremely challenging linguistic one, and is very sensitive to how questions are phrased; a set of rules that are effective at formulating subquestions from Jeopardy! clues may not be as effective for other types of questions. Multi-dimensional merging also requires that the question be divided up, but it does not require that the parts of the question form coherent subquestions, since it is performed *after* all of the linguistic analysis and comparison to evidence. In our implementation of multi-dimensional merging, we simply divide up the question into single terms.

We consider both corpus expansion and question decomposition as complementary to our approach. Both approaches are included in our baseline Jeopardy! system, and corpus expansion is included in our baseline medical system. The fact that our results show postive impact on effective question answering shows that multi-dimensional merging can add value to a system that already uses both corpus expansion and question decomposition techniques.

Conclusion and perspectives

We introduced a general framework for aggregating evidence from different passages retrieved for a candidate answer. Moreover, we introduced a novel set of features, multi-dimensional feature merger or MDM features, that fit this framework and significantly improve the performance of the current state-of-the-art QA system, Watson. However, our framework is general and not restricted to Watson. It may be employed in any QA system that captures how well retrieved passages match the question under consideration.

In this paper, we only considered merging evidence across passages and question terms. However, this may be easily extended to merging evidence across passage scorers. There might be value in considering how different passage scorers match supporting passages with candidate answers. Using our framework, all that is required is adding a new dimension: depth to the two-dimensional matrix M, thus giving rise to a $3-D$ matrix, say $M3D$. Each two dimensional matrix, M in $M3D$ belongs one passage scorer. Therefore, depth of $M3D$ is the number of passage scorers used to match supporting passages with the question. In the future, we will explore decomposing and thus deriving features from this $3-D$ matrix, possibly using Tensor algebra (Kolda and Bader, 2008).

References

Chu-Carroll, J., Fan, J., Boguraev, B. K., Carmel, D., Sheinwald, D., and Welty, C. A. (2012). Finding needles in the haystack: Search and candidate generation. *IBM Journal Research and Developement*, 56.

Felshin, B. K. . G. B. . S. (2005). Syntactic and semantic decomposition strategies for question answering from multiple resources. *AAAI*.

Ferrucci, D. A., Brown, E. W., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J. M., Schlaefer, N., and Welty, C. A. (2010). Building watson: An overview of the deepqa project. *AI Magazine*, 31:59–79.

Gondek, D. C., Lally, A., Kalyanpur, A., Murdock, J. W., Duboue, P. A., Zhang, L., Pan, Y., Qiu, Z. M., and Welty, C. A. (2012). A framework for merging and ranking of answers in deepqa. *IBM Journal Research and Developement*, 56.

Hall, M. A. (2000). Correlation-based feature selection for discrete and numeric class machine learning. *17th International Conference of Machine Learning (ICML)*, pages 359–366.

Kalyanpur, A., Patwardhan, S., Boguraev, B. K., Lally, A., and Chu-Carroll, J. (2012). Fact-based question decomposition in deepqa. *IBM Journal Research and Developement*, 56:13:1–13:11.

Lally, A., Prager, J. M., McCord, M. C., Boguraev, B. K., Patwardhan, S., Fan, J., Fodor, P., and Chu-Carroll, J. (2012). Question analysis: How watson reads a clue. *IBM Journal of Research and Development*, 56.

Maybury, M. T. (2004). *New Directions in Question-Answering*. Melno Park CA: American Association for Artificial Intelligence.

Murdock, J. W., Fan, J., Lally, A., Shima, H., and Boguraev, B. K. (2012a). Textual evidence gathering and analysis. *IBM Journal Research and Developement*, 56.

Murdock, J. W., Kalyanpur, A., Welty, C. A., Fan, J., Ferrucci, D. A., Gondek, D. C., Zhang, L., and Kanayama, H. (2012b). Typing candidate answers using type coercion. *IBM Journal Research and Developement*, 56.

Schlaefer, N., Chu-Carroll, J., Nyberg, E., Fan, J., Zadrozny, W., and Ferrucci, D. (2011). Statistical source expansion for question answering. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 345–354, New York, NY, USA. ACM.

Simmons, R. F. (1970). Natural language question-answering systems: 1969. *Commun. ACM*, 13:15–30.

Strzalkowski, T. and Harabagiu, S. (2006). *Advances in Open-Domain Question-Answering*. Berlin Germany: Springer-Verlag.

Voorhees, E. and Tice, D. (2000). Building a question answering test collection. *SIGIR*, pages 200–207.