Supporting Human Answers for Advice-Seeking Questions in CQA Sites

Liora Braunstain[1], Oren Kurland[1], David Carmel[2], Idan Szpektor[2], and Anna Shtok[1]

[1] Faculty of Industrial Engineering and Management, Technion, Haifa 32000, Israel
liorab@campus.technion.ac.il,kurland@ie.technion.ac.il,annabel@tx.technion.ac.il
[2] Yahoo Labs, Haifa 31905, Israel
{dcarmel,idan}@yahoo-inc.com

Abstract. In many questions in Community Question Answering sites users look for the advice or opinion of other users who might offer diverse perspectives on a topic at hand. The novel task we address is providing supportive evidence for human answers to such questions, which will potentially help the asker in choosing answers that fit her needs. We present a support retrieval model that ranks sentences from Wikipedia by their presumed support for a human answer. The model outperforms a state-of-the-art textual entailment system designed to infer factual claims from texts. An important aspect of the model is the integration of relevance oriented and support oriented features.

1 Introduction

Most questions posted on Community-based Question Answering (CQA) websites, such as Yahoo Answers, Answers.com and StackExchange, do not target simple facts such as "*what is Brad Pitt's height?*" or "*how far is the moon from earth?*". Instead, askers expect some human touch in the answers to their questions. Especially, many questions look for recommendations, suggestions and opinions, *e.g.* "*what are some good horror movies for Halloween?*", "*should you wear a jockstrap under swimsuit?*" or "*how can I start to learn web development?*". According to our analysis, based on editorial judgments of 12,000 Yahoo Answers questions, 70% of all questions are advice or opinion seeking questions.

Examining answers for such advice-seeking questions, we found that quite often answerers do not provide supportive evidence for their recommendation, and that answers usually represent diverse perspectives of the different answerers for the question at hand. For example, answerers may recommend different horror movies. Still, the asker would like to choose only one or two movies to watch, and without additional supportive evidence her decision may be non-trivial.

In this paper we assume that askers would be happy to receive additional information that will help them in choosing the best fit for their need from the various suggestions or opinions provided in the CQA answers. More formally, we propose the novel task of retrieving sentences from the Web that provide support to a given recommendation or opinion that is part of an answer in a CQA site.

We refer to the part of the answer (*e.g.*, a sentence) that contains a recommendation as a *subjective claim* about the need expressed in the question (*e.g.*, a call for advice). For a sentence to be considered as *supporting the claim*, it should be relevant to the content of the claim and provide some supporting information; *e.g.*, examples, statistics, or testimony [1]. More specifically, a supporting sentence is one whose acceptance is likely to raise the confidence in the claim.

While supporting sentences may be part of the same answer containing the claim, or found in other answers given for the same question, in this paper we are interested in retrieving sentences from other sources which may provide different perspectives on the claim compared to content on CQA sites. For example, for the question "*what are some good horror movies?*", a typical CQA answer could be "*The Shining is a great movie; I love watching it every year*". On the other hand, a supporting sentence from external sites may contain information such as "*...in 2006, the Shining made it into Ebert's series of "Great Movie" reviews...*". Specifically, we focus on retrieving supporting sentences from Wikipedia, although our methods can be largely applied to other Web sites.

We present a general scheme of *Learning to Rank for Support*, in which the retrieval algorithm is directly optimized for ranking sentences by presumed support. Our feature set includes both relevance-oriented features, such as textual similarity, and support-oriented features, such as sentiment matching and similarity with language-model-based support priors.

We experimented with a new dataset containing 40 subjective claims from the Movies category of Yahoo Answers. For each claim, sentences retrieved from Wikipedia using relevance estimates were manually evaluated for relevance and support. The evaluated benchmark was then used to train and test our model. The results demonstrate the merits of integrating relevance-based and support-based features for the support ranking task. Furthermore, our model substantially outperforms a state-of-the-art Textual Entailment system used for support ranking. This result emphasizes the difference between prior work on supporting objective claims and our task of supporting subjective recommendations.

2 Ranking Sentences by Support

Our goal is to devise a sentence retrieval method that ranks sentences by the level of *support* they provide to a given *subjective* claim. For example, the sentence "*movie X received the Oscar academy award for the best film*" would be considered as providing strong support to the claim "*X is a good movie*".

We confine our treatment of the sentence retrieval task to claims about a single entity c_e — *e.g.* the movie X in the example above — since often advice-seeking CQA questions are about entities such as restaurants, movies, singers and products. For sentence s to provide support for a given claim c, s must be relevant to c and especially to the entity c_e that c is about. Hence, our approach for support ranking is based on an initial relevance ranking of sentences (Sect. 2.1). Then, a set of features is used in a learning-to-rank method for re-ranking the top-retrieved sentences by their (presumed) support for c (Sect. 2.2).

2.1 Initial Relevance Ranking

Our first step is to rank sentences by their presumed relevance to claim c. Since these sentences are part of documents in a corpus D, we follow common practice in work on sentence retrieval [2] and first apply document retrieval with respect to c. Then, the sentences in the top ranked documents are ranked for relevance.

We assume that each document $d \in D$ is composed of a title, d_t, and a body, d_b. This is the case for Wikipedia, which is used in our experiment, as well as for most Web pages. The initial document retrieval, henceforth **InitDoc**, is based on the document score $S_{SDM}(c; d_b)$. This score is assigned to the body of document d with respect to the claim c by the state-of-the-art sequential dependence model (SDM) from the Markov Random Field framework [3]. For texts x and y,

$$S_{SDM}(x; y) \stackrel{def}{=} \lambda_T S_T(x; y) + \lambda_O S_O(x; y) + \lambda_U S_U(x; y); \quad (1)$$

$S_T(x; y)$, $S_O(x; y)$ and $S_U(x; y)$ are the (smoothed) log likelihood values of the appearances of unigrams, ordered bigrams and unordered bigrams, respectively, of tokens from x in y; λ_T, λ_U, and λ_O are free parameters whose values sum to 1. We further bias the initial document ranking in favor of documents whose titles contain c_e — the entity the claim is about. Specifically, d is ranked by:

$$S_{InitDoc}(c; d) \stackrel{def}{=} \alpha S(c_e; d_t) + (1 - \alpha) S_{SDM}(c; d_b); \quad (2)$$

$S(c_e; d_t)$ is the log of the Dirichlet smoothed maximum likelihood estimate, with respect to d's title, of the n-gram which constitutes the entity c_e [4]; smoothing is based on n-gram counts in the corpus[3]; α is a free parameter.

To estimate the relevance of sentence s to the claim c, we can measure their similarity using, again, the SDM model. We follow common practice in work on passage retrieval [2], and interpolate, using a parameter β, the claim-sentence similarity score with the retrieval score of document d which s is part of:

$$S_{InitSent}(c; s) \stackrel{def}{=} \beta S_{SDM}(c; s) + (1 - \beta) S_{InitDoc}(c; d). \quad (3)$$

Eq. 3 is used to rank the sentences in the top-N retrieved documents; N is a free parameter. The k most highly ranked sentences serve for $\mathcal{S}_{\text{init}}^{[k]}$ — the initial set of sentences to be ranked for support. Herein, **InitSent** denotes the sentence score assigned in Eq. 3 which is used to induce the initial sentence ranking.

2.2 Learning to Rank for Support

Next, we rank the sentences in $\mathcal{S}_{\text{init}}^{[k]}$ by the support they provide to the claim. To this end, we apply a learning-to-rank (LTR) approach [5] to construct a ranking function designed to optimize support. Specifically, we use a training set of claims, their respective sentences, and labels of the support level the sentences provide for the claims. Each pair of a claim and a sentence, (c, s), is represented as a feature vector. Below, we detail our feature set. In Sect. 3 we report the performance of three LTR methods applied with these features.

[3] All SDM scoring function components in Eq. 1 also use the logs of Dirichlet smoothed estimates [3]. The smoothing parameter, μ, is set to the same value for all estimates.

Language-Model Similarities. We use the initial retrieval scores, InitDoc (Eq. 2) and InitSent (Eq. 3), as relevance-estimate features. Additionally, we use several language-model-based similarity estimates. Let $p_{JM}^{[\psi]}(w|x)$ be the probability assigned to term w by a Jelinek-Mercer smoothed unigram language model induced from text x using the smoothing parameter ψ [4];[4] setting $\psi = 0$ amounts to the maximum likelihood estimate of w with respect to x. The similarity between texts x and y is estimated using the cross entropy, CE, between their induced language models: $sim_{LM}(x, y) \stackrel{def}{=} -CE\left(p_{JM}^{[0]}(\cdot|x) \,\middle|\middle|\, p_{JM}^{[\psi]}(\cdot|y)\right)$; higher values of CE correspond to reduced similarity.

We use the following similarity features: (i) **ClaimTitle**: between the claim and the document title ($sim_{LM}(c, d_t)$); (ii) **EntTitle**: between the entity and the document title ($sim_{LM}(c_e, d_t)$); (iii) **ClaimBody**: between the claim and the document body ($sim_{LM}(c, d_b)$); (iv) **EntBody**: between the entity and the document body ($sim_{LM}(s_e, d_b)$); (v) **ClaimSent**: between the claim and the sentence ($sim_{LM}(c, s)$); and, (vi) **EntSent**: between the entity and the sentence ($sim_{LM}(c_e, s)$). The entity is treated here as a bag of terms. These relevance-based similarity estimates, some of which are components of Eq. 2 and 3, are weighed by the learning to rank method with respect to support ranking rather than relevance ranking, which helps to avoid metric divergence issues [3].

Semantic Similarities. Both the claim c and the candidate support sentence s can be short. Thus, to address potential vocabulary mismatch issues in textual similarity estimation, we also use semantic-based similarity measures that utilize word embedding [6]. Specifically, we use the word vectors, of dimension 300, trained over a Google news dataset with Word2Vec[5]. Let $\boldsymbol{w}$ denote the embedding vector representing term w. We measure the extent to which the terms in s "cover" the terms in c by **MaxSemSim**: $\sum_{w \in c} \max_{w' \in s} \cos(\boldsymbol{w}, \boldsymbol{w'})$. Additionally, we measure the similarity between the centroids of the claim and the sentence (cf. [7]), **CentSemSim**: $\cos(\frac{1}{|c|}\sum_{w \in c} \boldsymbol{w}, \frac{1}{|s|}\sum_{w' \in s} \boldsymbol{w'})$; $|c|$ and $|s|$ are the number of terms in the claim and sentence, respectively.

Sentiment Features. As the claim c is assumed to be subjective, we make the premise that a relevant sentence s is likely to also support c if the same sentiment is expressed in c and s. We use the Stanford sentiment analyzer[6] [8], pre-trained with the Rotten Tomatoes movie reviews dataset [9]. This tool produces, for a given text, a probability distribution over a 1–5 sentiment scale; 1 stands for "very negative" and 5 stands for "very positive". As a sentiment similarity feature, **SentimentSim**, we use the Jensen Shannon (JS) divergence between the sentiment distributions for the claim and the sentence. Higher JS values correspond to lower similarity. Additionally, we compute **SentimentEnt**: the entropy of the sentiment distribution induced for s. This feature attests to the focus (or lack thereof) of the sentiment distribution induced from the sentence.

[4] Smoothing is performed using the term statistics in the document corpus D.

[5] `https://code.google.com/p/word2vec/`

[6] `http://nlp.stanford.edu/software/corenlp.shtml`

Quality-Oriented Language Models. In general, we expect to find differences between the language used to describe entities that are of "high quality" compared to those of "low quality". Still, to construct language models for such classes of entities, labeled examples are needed. This labeling is typically missing from most Web sources. Yet, for many domains there are sites that provide ratings for entities, *e.g.*, user feedback for local businesses in `yelp.com`. We propose to transfer such ratings to other sites as noisy quality labels. Specifically, our test claims are about movies, and the sentences ranked for support are extracted from Wikipedia which does not provide explicit ratings. Therefore, we automatically labeled each Wikipedia page about a movie with the 1-5 star grade review posted for this movie in IMDB[7] (if exists). Using this knowledge transfer, five unigram language models were induced, one per rating grade l. Specifically, all Wikipedia pages of movies with an IMDB review of a grade l were concatenated to yield the text: $Text_l$.[8] Then, for sentence s, the claim-independent features, denoted **Prior**-l, that correspond to quality levels $l \in \{1, \ldots, 5\}$ are: $\frac{sim_{LM}(s,Text_l)}{\sum_{l'=1}^{5} sim_{LM}(s,Text_{l'})}$.

Sentence Style. The **StopWords** feature is the fraction of terms in the sentence that are stop words. High occurrence of stop words potentially attests to rich use of language [10], and consequently, to sentence quality. Stop words are determined using the Stanford parser[9]. We also use the sentence length, **SentLength**, as a prior signal for sentence quality.

3 Empirical Evaluation

3.1 Dataset

There is no publicly available dataset for evaluating sentence ranking for support of subjective claims that originate from advice-seeking questions and corresponding answers. Hence, we created a novel dataset[10] as follows. Fifty subjective claims about movies, which serve as the entities c_e, were collected from Yahoo Answers[11] by scanning its movies category. We looked for advice-seeking questions, which are common in the movies category, and selected answers that contain at least one movie title. Each pair of a question and a movie title appearing in an answer for the question was transformed to a claim by manually reformulating the question into an affirmative form and inserting the entity (movie title) as the subject. For example, the question "*any good science fiction movies?*" and the movie title "*Tron*" was transformed to the claim "*Tron is a good science fiction movie*".

The corpus used for sentence retrieval is a dump of the movies category of Wikipedia from March 2015, which contains 111,164 Wikipedia pages. For each

[7] IMDB snapshot from 08/01/2014.
[8] The order of concatenation has no effect since unigram language models are used.
[9] `http://nlp.stanford.edu/software/lex-parser.shtml`
[10] Available at `http://iew3.technion.ac.il/~kurland/supportRanking`
[11] `answers.yahoo.com`

Table 1. Examples of claims and supporting and non-supporting sentences.

claim	The Pursuit of Happyness is one of the best inspirational movies
support	"The Pursuit of Happyness" is an unexceptional film with exceptional performances
non-support	The Pursuit of Happyness is a 2006 American biographical drama film based on Chris Gardner's nearly one-year struggle with homelessness
claim	The Godfather is one of the top movies of all times
support	Also in 2002, "The Godfather" was ranked the second best film of all time by Film4, after "Star Wars Episode V: The Empire Strikes Back"
non-support	The opening shot of the film is a long, slow pullback, starting with a close-up of Bonasera, who is petitioning Don Corleone, and ending with the Godfather, seen from behind, framing the picture
claim	Saving private Ryan is a favourite war movie
support	In 2014, "Saving Private Ryan" was selected for preservation in the National Film Registry as per being deemed "culturally, historically, or aesthetically significant"
non-support	Saving Private Ryan was released on home video in May 1999, earning $44 million from sales

claim, 100 sentences were retrieved using the initial sentence retrieval approach, InitSent (Sect. 2.1). Each of these 100 sentences was categorized by five annotators from CrowdFlower[12] into: (1) not relevant to the claim, (2) strong non-support, (3) medium non-support, (4) neutral, (5) medium support, (6) strong support. The final label was determined by a majority vote.

We used the following induced scales: (a) ***binary relevance***: not relevant (category 1) vs. relevant (categories 2-6); (b) ***binary support***: non-support (categories 1-4) vs. support (categories 5-6); (c) ***graded support***: non-support (categories 1-4), weak support (category 5) and strong support (category 6). The Fleiss' Kappa inter-annotator agreement rates are: 0.68 (substantial) for *binary relevance*, 0.592 (moderate) for *binary support* and 0.457 (moderate) for *graded support*. Table 1 provides examples of claims and relevant (on a binary scale) sentences that either support the claim or not (i.e., binary support scale is used).

Ten out of the fifty claims had no support sentences and were not used for evaluation. For the forty remaining claims, on average, half of the support sentences were weak support and the other half were strong support. On average, 23.5% of the relevant sentences are supportive (binary scale). The median, average and standard deviation of relevant sentences, and of support sentences (binary scale), per claim are: 29, 40.5, 29.3 and 5.5, 7.4 and 6.7, respectively.

3.2 Methods

For the learning-to-rank methods (LTR) we used a linear SVMrank (LinearSVM) [11], a second-degree polynomial kernel SVMrank (PolySVM) [11], and LambdaMART [12], which is a state-of-the-art learning-to-rank method [5]. We used the LTR methods[13] with all the features described in Sect. 2.2 for ranking sentences by support and by relevance — i.e., we either optimized performance for

[12] `www.crowdflower.com`

[13] The implementations of LinearSVM and PolySVM are from `http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`. The LambdaMART implementation is from `http://sourceforge.net/p/lemur/wiki/RankLib/`. All methods are used with default free-parameter values of the corresponding implementations.

Table 2. Main result table. Comparing the relevance-ranking and support-ranking performance of the three LTR methods with that of the initial sentence ranking (InitSent). Boldface: the best result in a column; 'i', 'l' and 'p' mark statistically significant differences with InitSent, LinearSVM and PolySVM, respectively.

	relevance				support			
	NDCG@1	NDCG@3	NDCG@10	p@5	NDCG@1	NDCG@3	NDCG@10	p@5
InitSent	.775	.766	.739	.730	.083	.165	.295	.215
LinearSVM	.800	.786	.782	.770	$.441^{i}$	$.478^{i}$	$.519^{i}$	$.410^{i}$
PolySVM	.800	.839	$\mathbf{.852}^{i,l}$	$\mathbf{.835}^{i}$	$.525^{i}$	$.527^{i}$	$.564^{i,l}$	$.445^{i}$
LambdaMART	**.825**	**.844**	.808	$\mathbf{.835}^{i}$	$\mathbf{.608}^{i}$	$\mathbf{.540}^{i}$	$\mathbf{.593}^{i}$	$\mathbf{.515}^{i,l}_{p}$

support or for relevance. Leave-one-out cross validation, performed over queries, was used for training and testing.

The Indri toolkit[14] was used for experiments. Krovetz stemming was applied to claims and sentences only for inducing the initial document and sentence ranking and for computing the language-model-based similarity features described in Sect. 2.2. For these features, stopwords on the INQUERY list were removed only from claims. The number of documents (Wikipedia pages) initially retrieved using InitDoc (Eq. 2) for each claim was $N = 1000$; α was set to 0.66 to boost the ranking of the Wikipedia page about the target movie in the claim. Then, $k = 100$ sentences from these 1000 documents were retrieved using InitSent (Eq. 3) with $\beta = 0.5$. These 100 sentences constitute the set $\mathcal{S}_{\text{init}}^{[100]}$ which is re-ranked by the LTR methods. The SDM free parameters, λ_T, λ_O and λ_U were automatically set, in both InitDoc and InitSent, using the approach proposed in [13]. For language models, the Dirichlet smoothing parameter, μ, and the Jelinek-Mercer smoothing parameter, ψ, were set to the standard values of 1000 and 0.1, respectively [4]. We note that the free parameters of the initial document and sentence ranking could not be set using training data, as such data is only available for the initially retrieved sentence set, $\mathcal{S}_{\text{init}}^{[100]}$, as described above.

We view support ranking as a high-precision oriented task in which users are interested in seeing a few sentences that strongly support the claims at hand. Hence, for evaluation measures we use NDCG@1, NDCG@3, NDCG@10 and the precision of the top-5 sentences (p@5). The NDCG performance numbers for support ranking are based on *graded support* scale, and those for p@5 are based on the *binary support* scale. All performance numbers for relevance ranking are based on the *binary relevance* scale. LambdaMART was trained for NDCG@10 as this yielded, in general, better support ranking performance across the evaluation measures than using NDCG@1 or NDCG@3. Statistically significant differences of performance are determined using the two tailed paired t-test with $p = 0.05$.

3.3 Results

Table 2 presents our main results. We see that all three LTR methods outperform the initial sentence ranking, InitSent, in terms of relevance ranking. Although few

[14] `www.lemurproject.org`

Table 3. Comparison and integration with a state-of-the-art textual entailment algorithm (P1EDA). LMart stands for "LambdaMart". Boldface: the best result in a column. Statistically significant differences with InitSent, P1EDA and LMart are marked with 'i', 'p', and 'm', respectively.

	relevance				support			
	NDCG@1	NDCG@3	NDCG@10	p@5	NDCG@1	NDCG@3	NDCG@10	p@5
InitSent	.775	.766	.739	.730	.083	.165	.295	.215
P1EDA	$.525^{i}$	$.496^{i}$	$.462^{i}$	$.475^{i}$	.066	.093	$.129^{i}$	$.120^{i}$
LMart	$.825^{i,p}$	$\mathbf{.844}^{i,p}$	$.808^{i,p}$	$\mathbf{.835}^{i,p}$	$\mathbf{.608}^{i,p}$	$.540^{i,p}$	$.593^{i,p}$	$\mathbf{.515}^{i,p}$
LMart+P1EDA	$\mathbf{.850}^{p}$	$.836^{p}$	$\mathbf{.811}^{p}$	$.815^{p,m}$	$.600^{i,p}$	$\mathbf{.571}^{i,p}$	$\mathbf{.609}^{i,p}$	$.490^{i,p}$

of these improvements are statistically significant, they attest to the potential merits of using the additional relevance-based features described in Sect. 2.2. More importantly, all LTR methods substantially, and statistically significantly, outperform the initial (relevance-based) sentence ranking in terms of support. This result emphasizes the difference between relevance and support and shows that our proposed features for support ranking are quite effective, especially when used in a non-linear ranker such as LambdaMART.

In Sect. 1 and 4 we discuss the difference between subjective and factoid claims. To further explore this difference, we compare our best performing LambdaMART method with the P1EDA[15] state-of-the-art textual entailment algorithm [14] when both are used for the support-ranking task we address here. P1EDA was designed for factual claims[16]. Specifically, given a claim and a candidate sentence, P1EDA produces a classification decision of whether the sentence entails the claim, accompanied with a confidence level. The confidence level was used for support (and relevance) ranking. We also tested the inclusion of P1EDA's output (confidence level) as an additional feature in LambdaMART, yielding the LMart+P1EDA method. Table 3 depicts the performance numbers.

We can see in Table 3 that P1EDA is (substantially) outperformed by both InitSent and LambdaMART, for both relevance and support ranking. Since the claims in our setting are simple, this finding implies that approaches for identifying texts that support (or "prove") a factoid claim may not be effective for the task of supporting subjective claims. The integration of P1EDA as a feature in LambdaMART improves performance (although not to a statistically significant degree) for some of the evaluation measures, including NDCG@10 for which the ranker was trained, and hurts performance for others — statistically significantly so in only a single case[17].

Integrating P1EDA with *only* our semantic-similarity features using LambdaMART, which is a conceptually similar approach to a classification method employed in some work on argument mining [16], resulted in support-ranking performance that is substantially worse than that of using all our proposed features in LambdaMART. Actual numbers are omitted due to space considerations and as they convey no additional insight.

[15] `http://hltfbk.github.io/Excitement-Open-Platform/`

[16] We trained P1EDA using the SNLI data set [15] , which contains 549,366 examples.

[17] Integrating P1EDA in PolySVM did not yield support-ranking improvements.

Table 4. Using features alone (specifically, the 10 that yield the highest NDCG@10 support ranking) to rank the initial sentence list vs. integrating all features in LambdaMART. Boldface: the best result in a column; 'm': statistically significant difference with LambdaMART.

	NDCG@1		NDCG@3		NDCG@10	
	relevance	support	relevance	support	relevance	support
LambdaMART	**.825**	**.608**	**.844**	**.540**	.808	**.593**
ClaimTitle	.700	$.191^m$	.754	$.216^m$	.796	$.302^m$
EntTitle	.725	$.200^m$	.791	$.212^m$	**.811**	$.297^m$
InitSent	.775	$.083^m$	.766	$.165^m$	.739	$.295^m$
SentimentSim	$.475^m$	$.225^m$	$.460^m$	$.239^m$	$.433^m$	$.295^m$
InitDoc	$.600^m$	$.150^m$	$.669^m$	$.208^m$	$.684^m$	$.277^m$
ClaimSent	$.575^m$	$.291^m$	$.614^m$	$.272^m$	$.603^m$	$.276^m$
MaxSemSim	.700	$.183^m$	$.688^m$	$.195^m$	$.654^m$	$.228^m$
ClaimBody	$.450^m$	$.041^m$	$.612^m$	$.130^m$	$.696^m$	$.222^m$
Prior-5	$.500^m$	$.083^m$	$.479^m$	$.142^m$	$.506^m$	$.221^m$
EntSent	.650	$.066^m$	$.667^m$	$.113^m$	$.692^m$	$.188^m$

Feature Analysis. To analyze the contribution of individual features to overall performance, Table 4 compares LambdaMART, used with all features, to using individual features alone for ranking. As LambdaMART was trained for NDCG@10 for support ranking, we explore the 10 features that yielded the highest NDCG@10 support-ranking performance.

Table 4 clearly shows that while a few features yield support-ranking performance that transcends that of the initial sentence ranking (InitSent), LambdaMART that integrates all features yields substantially, and statistically significantly, better support-ranking performance. This finding attests to the importance of integrating various features for support ranking. LambdaMART is also superior to almost all ten features for relevance ranking[18].

We see that quite a few of the top-10 features are (lexical) similarities between the claim and/or the entity it is about and the sentence and/or its ambient document. This shows that (direct) estimates of claim-sentence relevance can be quite important for support ranking, as is expected. Yet, integrating these estimates with support-oriented estimates is important for attaining highly effective support ranking performance as is evident in LambdaMART's performance.

SentimentSim, the sentiment similarity between the claim and the sentence, is among the most effective features when used alone for support ranking. Additional ablation tests[19] reveal that removing SentimentSim from the set of all features results in the most severe performance degradation for all three learning-to-rank methods. Indeed, sentiment is an important aspect of subjective claims, and therefore, of inferring support for these claims.

We also found that ranking sentences by decreasing entropy of sentiment (SentimentEnt) is superior to ranking by increasing entropy for NDCG@1 and NDCG@3, while for NDCG@10 the reverse holds. The former finding is a conceptual reminiscent of those about using the entropy of a document term distri-

[18] For relevance ranking, LambdaMART was trained for *binary relevance*.

[19] Actual numbers are omitted due to space considerations and as they convey no additional insight.

bution for the document prior for Web search [10]: the higher the entropy, the "broader" the textual unit is – in our case, in terms of expressed sentiment — which presumably implies to a higher prior.

Finally, Table 4 also shows that Prior-5 is the most effective claim-independent feature[20]. It is the similarity between a language model of the sentence and that induced from Wikipedia movie pages which received high grade (5 stars) reviews in IMDB. This shows that although Wikipedia authors aim to be objective in their writing, the style and information for high rated movies is still quite different from that for lower rated ones, and it can potentially be modeled via the automatic knowledge transfer and labeling method proposed in Sect. 2.2.

4 Related Work

A few lines of research are related to our work. The Textual Entailment task is inferring the truthfulness of a textual statement (*hypothesis*) from a given text [17]. A more specific incarnation of Textual Inference is automatic Question Answering (QA). Work on these tasks focused on factoid claims for which a clear correct/incorrect labeling should be inferred from supportive evidence. Thus, typical textual inference approaches are designed to find the claim (*e.g.* a candidate answer in QA) embedded in the supporting text, although it may be rephrased. In contrast, in this paper, claims originate from CQA users who provide subjective recommendations rather than state facts. Our model, designed for ranking sentences by support for a subjective claim, significantly outperforms for this task a state-of-the-art textual entailment method as shown in Sect. 3.3.

Blanco and Zaragoza [18] introduce methods for retrieving sentences that explain the relationship between a Web query and a related named entity, as part of the *entity ranking* task. In contrast, we rank sentences by support for a subjective claim. Kim et al. [19] present methods for retrieving sentences that explain reasons for sentiment expressed about an aspect of a topic. In contrast to these sentence ranking methods [18, 19], ours utilizes a learning-to-rank method that integrates various relevance and support features not used in [18, 19].

The task most related to ours is *argument mining* (e.g., [20, 16, 21–24]). Specifically, arguments supporting or contradicting a claim about a given debatable (often controversial) topic are sought. Some of the *types* of features we use for support ranking have also been used for argument mining; namely, semantic [16, 24] and sentiment [24] similarities between the claim and a candidate argument. Yet, the actual estimates and techniques used here to induce these features are different than those in work on argument mining [16, 24]. Furthermore, the knowledge-transfer-based features we utilize, and whose effectiveness was demonstrated in Sect. 3.3, are novel to this study.

Interestingly, while textual entailment features were found to be effective for argument mining [20, 16], this is not the case for support ranking (see Sect. 3.3). This finding could be attributed to the fundamentally different nature of claims

[20] Ablation tests reveal that removing this feature results in the second most substantial decrease of support-ranking performance among all features.

used in our work, and those used in argument mining. That is, our claims originate from answers to advice-seeking questions of subjective nature, rather than being about a given debatable/controversial topic. Also, additional information about the debatable topic which was utilized in work on argument mining [24] is not available in our setting.

Often, work on argument mining [24], similarly to that on question answering (e.g., [25]), focuses on finding supporting or contradicting evidence in the same document in which the claim appears. In contrast, we retrieve supporting sentences from the Web for claims originating from CQA sites. In fact, there has been very little work on using sentence retrieval for argument mining [22]. In contrast to our work, a Boolean retrieval method was used, different features were utilized, and relevance-based estimates were not integrated with support-based estimates using a learning-to-rank approach.

5 Conclusions and Future Work

We addressed a novel task: ranking sentences from the Web by the support they provide to a subjective claim. The claim originates from an answer provided in a community question answering (CQA) site to an advice-seeking question.

Our support-ranking model utilizes various features in a learning-to-rank method; some are relevance oriented while others are support oriented. Empirical evaluation performed using a new dataset of claims created from Yahoo Answers attested to the merits of our proposed approach.

For future work we intend to extend the set of features, explore additional data domains, and study the utilization of supportive sentences in answers posted for subjective questions in CQA sites.

Acknowledgments. We thank the reviewers for their helpful comments, and Omer Levy and Vered Shwartz for their help with the textual entailment tool used for experiments. This work was supported in part by a Yahoo! faculty research and engagement award.

References

1. Rieke, R., Sillars, M.: Argumentation and Critical Decision Making. Longman Series in Rhetoric and Society. Longman (1997)
2. Murdock, V.: Exploring sentence retrieval. VDM Verlag (2008)
3. Metzler, D., Croft, W.B.: A Markov random field model for term dependencies. In: Proc. of SIGIR. (2005) 472–479
4. Zhai, C., Lafferty, J.D.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proc. of SIGIR. (2001) 334–342
5. Liu, T.Y.: Learning to rank for information retrieval. Foundations and Trends in Information Retrieval **3**(3) (2009)
6. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proc. of NIPS. (2013) 3111–3119

7. Vulic, I., Moens, M.: Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In: Proc. of SIGIR. (2015) 363–372
8. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C., Ng, A., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proc. of EMNLP. (2013) 363–372
9. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval **2**(1-2) (2007) 1–135
10. Bendersky, M., Croft, W.B., Diao, Y.: Quality-biased ranking of web documents. In: Proc. of WSDM. (2011) 95–104
11. Joachims, T.: Training linear SVMs in linear time. In: Proc. of KDD. (2006) 217–226
12. Burges, C.J.: From RankNet to LambdaRank to LambdaMART: An overview. Technical report, Microsoft Research (2010)
13. Zhou, Y., Croft, B.: Query performance prediction in web search environments. In: Proc. of SIGIR. (2007) 543–550
14. Noh, T.G., Pado, S., Shwartz, V., Dagan, I., Nastase, V., Eichler, K., Kotlerman, L., Adler, M.: Multi-level alignments as an extensible representation basis for textual entailment algorithms. In: Proc. of SEM. (2015)
15. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A large annotated corpus for learning natural language inference. In: Proc. of EMNLP. (2015) 632–642
16. Boltužić, F., Šnajder, J.: Back up your stance: Recognizing arguments in online discussions. In: Proc. of the First Workshop on Argumentation Mining, Association for Computational Linguistics (2014) 49–58
17. Dagan, I., Roth, D., Sammons, M., Zanzotto, F.M.: Recognizing textual entailment: Models and applications. Synthesis Lectures on Human Language Technologies **6**(4) (2013) 1–220
18. Blanco, R., Zaragoza, H.: Finding support sentences for entities. In: Proc. of SIGIR. (2010) 339–346
19. Kim, H.D., Castellanos, M., Hsu, M., Zhai, C., Dayal, U., Ghosh, R.: Ranking explanatory sentences for opinion summarization. In: Proc. of SIGIR. (2013) 1069–1072
20. Cabrio, E., Villata, S.: Combining textual entailment and argumentation theory for supporting online debates interactions. In: Proc. of ACL. (2012) 208–212
21. Green, N., Ashley, K., Litman, D., Reed, C., Walker, V., eds.: Proc. of the First Workshop on Argumentation Mining. ACL (2014)
22. Sato, M., Yanai, K., Yanase, T., Miyoshi, T., Iwayama, M., Sun, Q., Niwa, Y.: End-to-end argument generation system in debating. In: Proc. of ACL-IJCNLP 2015 System Demonstrations. (2015)
23. : Proc. of the second workshop on argumentation mining (NAACL) (2015)
24. Rinott, R., Dankin, L., Alzate Perez, C., Khapra, M.M., Aharoni, E., Slonim, N.: Show me your evidence - an automatic method for context dependent evidence detection. In: Proc. of EMNLP. (2015) 440–450
25. Brill, E., Lin, J.J., Banko, M., Dumais, S.T., Ng, A.Y., et al.: Data-intensive question answering. In: Proc. of TREC. Volume 56. (2001) 90