Bayesian Supervised Domain Adaptation for Short Text Similarity

Md Arafat Sultan[1,2] **Jordan Boyd-Graber**[2] **Tamara Sumner**[1,2]
[1]Institute of Cognitive Science
[2]Department of Computer Science
University of Colorado, Boulder, CO
{arafat.sultan,Jordan.Boyd.Graber,sumner}@colorado.edu

Abstract

Identification of short text similarity (STS) is a high-utility NLP task with applications in a variety of domains. We explore adaptation of STS algorithms to different target domains and applications. A two-level hierarchical Bayesian model is employed for domain adaptation (DA) of a linear STS model to text from different sources (e.g., news, tweets). This model is then further extended for multitask learning (MTL) of three related tasks: STS, short answer scoring (SAS) and answer sentence ranking (ASR). In our experiments, the adaptive model demonstrates better overall cross-domain and cross-task performance over two non-adaptive baselines.

1 Short Text Similarity: The Need for Domain Adaptation

Given two snippets of text—neither longer than a few sentences—short text similarity (**STS**) determines how semantically close they are. STS has a broad range of applications: question answering (Yao et al., 2013; Severyn and Moschitti, 2015), text summarization (Dasgupta et al., 2013; Wang et al., 2013), machine translation evaluation (Chan and Ng, 2008; Liu et al., 2011), and grading of student answers in academic tests (Mohler et al., 2011; Ramachandran et al., 2015).

STS is typically viewed as a *supervised* machine learning problem (Bär et al., 2012; Lynum et al., 2014; Hänig et al., 2015). SemEval contests (Agirre et al., 2012; Agirre et al., 2015) have spurred recent progress in STS and have provided valuable training data for these supervised approaches. However, similarity varies across domains, as does the underlying text; e.g., syntactically well-formed academic text versus informal English in forum QA.

Our goal is to effectively use domain adaptation (DA) to transfer information from these disparate STS domains. While "domain" can take a range of meanings, we consider adaptation to different (1) sources of text (e.g., news headlines, tweets), and (2) applications of STS (e.g., QA vs. answer grading). Our goal is to improve performance in a new domain with few in-domain annotations by using many out-of-domain ones (Section 2).

In Section 3, we describe our Bayesian approach that posits that per-domain parameter vectors share a common Gaussian prior that represents the global parameter vector. Importantly, this idea can be extended with little effort to a nested domain hierarchy (domains within domains), which allows us to create a single, unified STS model that *generalizes across domains as well as tasks*, capturing the nuances that an STS system must have for tasks such as short answer scoring or question answering.

We compare our DA methods against two baselines: (1) a domain-agnostic model that uses all training data and does not distinguish between in-domain and out-of-domain examples, and (2) a model that learns only from in-domain examples. Section 5 shows that across ten different STS domains, the adaptive model consistently outperforms the first baseline while performing at least as well as the second across training datasets of different sizes. Our multitask model also yields better overall results over the same baselines across three related tasks: (1) STS, (2) short answer scoring (SAS), and (3) answer sentence ranking (ASR) for question answering.

Proceedings of NAACL-HLT 2016, pages 927–936,
San Diego, California, June 12-17, 2016.

2 Tasks and Datasets

Short Text Similarity (STS) Given two short texts, STS provides a real-valued score that represents their degree of semantic similarity. Our STS datasets come from the SemEval 2012–2015 corpora, containing over 14,000 human-annotated sentence pairs (via Amazon Mechanical Turk) from domains like news, tweets, forum posts, and image descriptions.

For our experiments, we select ten datasets from ten different domains, containing 6,450 sentence pairs.[1] This selection is intended to maximize (a) the number of domains, (b) domain uniqueness: of three different news headlines datasets, for example, we select the most recent (2015), discarding older ones (2013, 2014), and (c) amount of per-domain data available: we exclude the FNWN (2013) dataset with 189 annotations, for example, because it limits per-domain training data in our experiments. Sizes of the selected datasets range from 375 to 750 pairs. Average correlation (Pearson's r) among annotators ranges from 58.6% to 88.8% on individual datasets (above 70% for most) (Agirre et al., 2012; Agirre et al., 2013; Agirre et al., 2014; Agirre et al., 2015).

Short Answer Scoring (SAS) SAS comes in different forms; we explore a form where for a short-answer question, a gold answer is provided, and the goal is to grade student answers based on how similar they are to the gold answer (Ramachandran et al., 2015). We use a dataset of undergraduate data structures questions and student responses graded by two judges (Mohler et al., 2011). These questions are spread across ten different assignments and two examinations, each on a related set of topics (e.g., programming basics, sorting algorithms). Inter-annotator agreement is 58.6% (Pearson's ρ) and 0.659 (RMSE on a 5-point scale). We discard assignments with fewer than 200 pairs, retaining 1,182 student responses to forty questions spread across five assignments and tests.[2]

Answer Sentence Ranking (ASR) Given a factoid question and a set of candidate answer sentences, ASR orders candidates so that sentences containing the answer are ranked higher. Text similarity is the foundation of most prior work: a candidate sentence's relevance is based on its similarity with the question (Wang et al., 2007; Yao et al., 2013; Severyn and Moschitti, 2015).

For our ASR experiments, we use factoid questions developed by Wang et al. (2007) from Text REtrieval Conferences (TREC) 8–13. Candidate QA pairs of a question and a candidate were labeled with whether the candidate answers the question. The questions are of different types (e.g., *what*, *where*); we retain 2,247 QA pairs under four question types, each with at least 200 answer candidates in the combined development and test sets.[3] Each question type represents a unique topical domain—*who* questions are about persons and *how many* questions are about quantities.

3 Bayesian Domain Adaptation for STS

We first discuss our base linear models for the three tasks: Bayesian L_2-regularized linear (for STS and SAS) and logistic (for ASR) regression. We extend these models for (1) adaptation across different short text similarity domains, and (2) multitask learning of short text similarity (STS), short answer scoring (SAS), and answer sentence ranking (ASR).

3.1 Base Models

In our base models (Figure 1), the feature vector $\boldsymbol{f}$ combines with the feature weight vector $\boldsymbol{w}$ (including a bias term w_0) to form predictions. Each parameter $w_i \in \boldsymbol{w}$ has its own zero-mean Gaussian prior with its standard deviation σ_{w_i} distributed uniformly in $[0, m_{\sigma_w}]$, the covariance matrix $\boldsymbol{\Sigma}_w$ is diagonal, and the zero-mean prior L_2 regularizes the model.

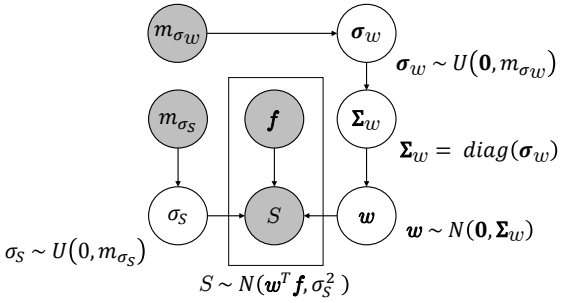
In the linear model (Figure 1a), S is the output (similarity score for STS; answer score for SAS) and is normally distributed around the dot product $\boldsymbol{w}^T\boldsymbol{f}$. The model error σ_S has a uniform prior over a pre-specified range $[0, m_{\sigma_S}]$. In the logistic model (Figure 1b) for ASR, the probability p that the candidate sentence answers the question, is (1) the sigmoid of $\boldsymbol{w}^T\boldsymbol{f}$, and (2) the Bernoulli prior of A, whether or not the candidate answers the question.

The common vectors $\boldsymbol{w}$ and $\boldsymbol{f}$ in these models enable joint parameter learning and consequently multitask learning (Section 3.3).

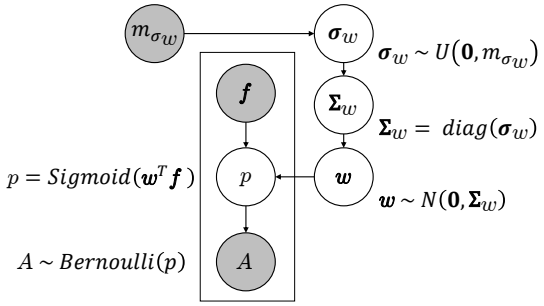
[1] 2012: MSRpar-test; 2013: SMT; 2014: Deft-forum, OnWN, Tweet-news; 2015: Answers-forums, Answers-students, Belief, Headlines and Images.

[2] Assignments: #1, #2, and #3; Exams: #11 and #12.

[3] *what*, *when*, *who* and *how many*.



(a) Bayesian ridge regression for STS and SAS.



(b) Bayesian logistic regression for ASR.

Figure 1: Base models for STS, SAS and ASR. Plates represent replication across sentence pairs. Each model learns weight vector $\boldsymbol{w}$. For STS and SAS, the real-valued output S (similarity or student score) is normally distributed around the weight-feature dot product $\boldsymbol{w}^T \boldsymbol{f}$. For ASR, the sigmoid of this dot product is the Bernoulli prior for the binary output A, relevance of the question's answer candidate.

3.2 Adaptation to STS Domains

Domain adaptation for the linear model (Figure 1a) learns a separate weight vector $\boldsymbol{w}_d$ for each domain d (i.e., applied to similarity computations for test pairs in domain d) alongside a common, global domain-agnostic weight vector $\boldsymbol{w}_*$, which has a zero-mean Gaussian prior and serves as the Gaussian prior mean for each $\boldsymbol{w}_d$. Figure 2 shows the model. Both $\boldsymbol{w}_*$ and $\boldsymbol{w}_d$ have hyperpriors identical to $\boldsymbol{w}$ in Figure 1a.[4]

Each $\boldsymbol{w}_d$ depends not just on its domain-specific observations but also on information derived from the global, shared parameter $\boldsymbol{w}_*$. The balance between capturing in-domain information and inductive transfer is regulated by Σ_w; larger variance allows $\boldsymbol{w}_d$ more freedom to reflect the domain.

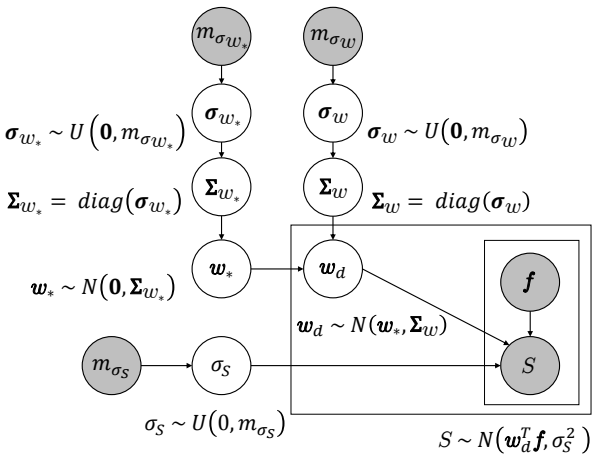


Figure 2: Adaptation to different STS domains. The outer plate represents replication across domains. Joint learning of a global weight vector $\boldsymbol{w}_*$ along with individual domain-specific vectors $\boldsymbol{w}_d$ enables inductive transfer among domains.

3.3 Multitask Learning

An advantage of hierarchical DA is that it extends easily to arbitrarily nested domains. Our multitask learning model (Figure 3) models topical domains nested within one of three related tasks: STS, SAS, and ASR (Section 2). This model adds a level to the hierarchy of weight vectors: each domain-level $\boldsymbol{w}_d$ is now normally distributed around a task-level weight vector (e.g., $\boldsymbol{w}_{\text{STS}}$), which in turn has global Gaussian mean $\boldsymbol{w}_*$.[5] Like the DA model, all weights in the same level share common variance hyperparameters while those across different levels are separate.

Again, this hierarchical structure (1) jointly learns global, task-level and domain-level feature weights enabling inductive transfer among tasks and domains while (2) retaining the distinction between in-domain and out-of-domain annotations. A task-specific model (Figure 1) that only learns from in-domain annotations supports only (2). On the other hand, a non-hierarchical joint model (Figure 4) supports only (1): it learns a single shared $\boldsymbol{w}$ applied to any test pair regardless of task or domain. We compare these models in Section 5.

[4] Results do not improve with individual domain-specific instances of σ_S and $\boldsymbol{\sigma}_w$, consistent with Finkel and Manning (2009) for dependency parsing and named entity recognition.

[5] We use the same variable for the domain-specific parameter $\boldsymbol{w}_d$ across tasks to simplify notation.

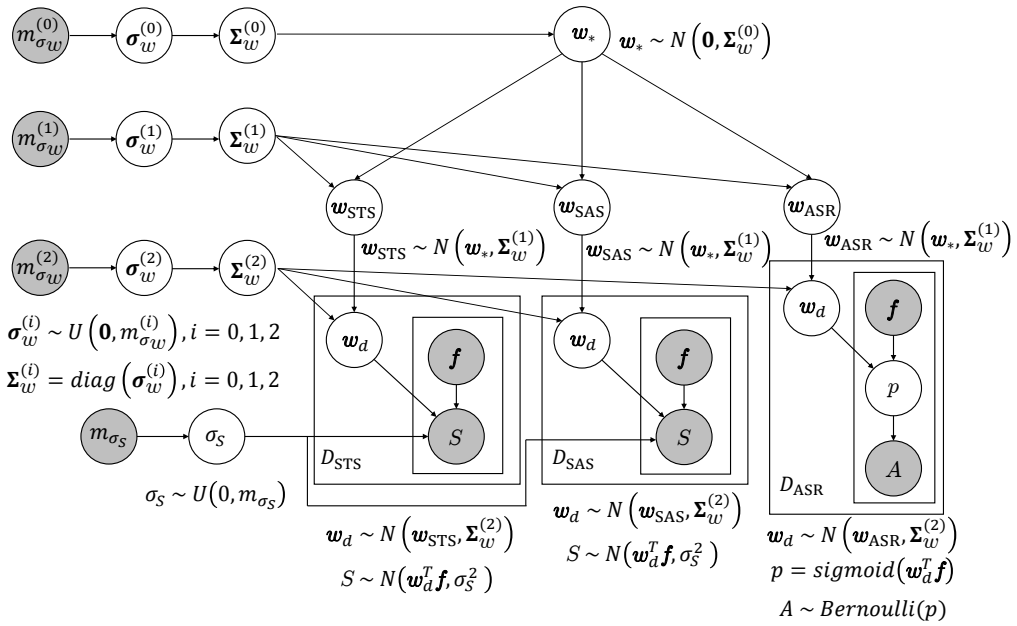


Figure 3: Multitask learning: STS, SAS and ASR. Global ($\boldsymbol{w}_*$), task-specific ($\boldsymbol{w}_{\text{STS}}$, $\boldsymbol{w}_{\text{SAS}}$, $\boldsymbol{w}_{\text{ASR}}$) and domain-specific ($\boldsymbol{w}_d$) weight vectors are jointly learned, enabling transfer across domains and tasks.

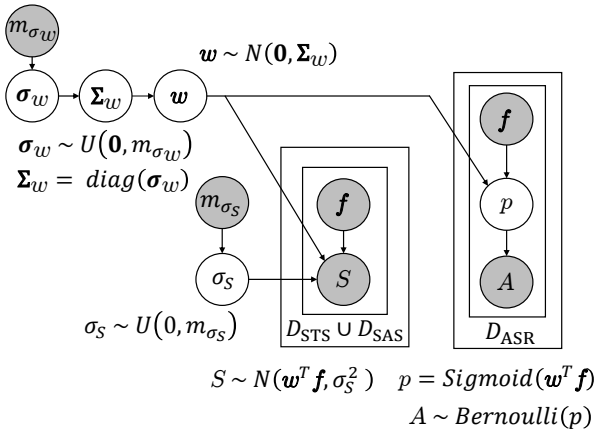


Figure 4: A non-hierarchical joint model for STS, SAS and ASR. A common weight vector $\boldsymbol{w}$ is learned for all tasks and domains.

4 Features

Any feature-based STS model can serve as the base model for a hierarchical Bayesian adaptation framework. For our experiments, we adopt the feature set of the ridge regression model in Sultan et al. (2015), the best-performing system at SemEval-2015 (Agirre et al., 2015).

Input sentences $S^{(1)} = (w_1^{(1)}, ..., w_n^{(1)})$ and $S^{(2)} = (w_1^{(2)}, ..., w_m^{(2)})$ (where each w is a token) produce two similarity features. The first is the proportion of content words in $S^{(1)}$ and $S^{(2)}$ (combined) that have a semantically similar word—identified using a monolingual word aligner (Sultan et al., 2014)—in the other sentence. The overall semantic similarity of a word pair $(w_i^{(1)}, w_j^{(2)}) \in S^{(1)} \times S^{(2)}$ is a weighted sum of lexical and contextual similarities: a paraphrase database (Ganitkevitch et al., 2013, PPDB) identifies lexically similar words; contextual similarity is the average lexical similarity in (1) dependencies of $w_i^{(1)}$ in $S^{(1)}$ and $w_j^{(2)}$ in $S^{(2)}$, and (2) content words in [-3, 3] windows around $w_i^{(1)}$ in $S^{(1)}$ and $w_j^{(2)}$ in $S^{(2)}$. Lexical similarity scores of pairs in PPDB as well as weights of word and contextual similarities are optimized on an alignment dataset (Brockett, 2007). To avoid penalizing long answer snippets (that still have the desired semantic content) in SAS and ASR, word alignment proportions outside the reference (gold) answer (SAS) and the question (ASR) are ignored.

The second feature captures finer-grained similarities between related words (e.g., `cell` and `organism`). Given the 400-dimensional embedding (Baroni et al., 2014) of each content word (lemmatized) in an input sentence, we compute a sentence vector by adding its content lemma vectors. The co-

Task	Current SOA	Our Model
STS	Pearson's r = 73.6%	Pearson's r = 73.7%
SAS	Pearson's r = 51.8% RMSE = 19.6%	Pearson's r = 56.4% RMSE = 18.1%
ASR	MAP = 74.6% MRR = 80.8%	MAP = 76.0% MRR = 82.8%

Table 1: Our base linear models beat the state of the art in STS, SAS and ASR.

sine similarity between the $S^{(1)}$ and $S^{(2)}$ vectors is then used as an STS feature. Baroni et al. develop the word embeddings using `word2vec`[6] from a corpus of about 2.8 billion tokens, using the Continuous Bag-of-Words (CBOW) model proposed by Mikolov et al. (2013).

5 Experiments

For each of the three tasks, we first assess the performance of our base model to (1) verify our sampling-based Bayesian implementations, and (2) compare to the state of the art. We train each model with a Metropolis-within-Gibbs sampler with 50,000 samples using PyMC (Patil et al., 2010; Salvatier et al., 2015), discarding the first half of the samples as burn-in. The variances m_{σ_w} and m_{σ_S} are both set to 100. Base models are evaluated on the entire test set for each task, and the same training examples as in the state-of-the-art systems are used. Table 1 shows the results.

Following SemEval, we report a weighted sum of correlations (Pearson's r) across all test sets for STS, where the weight of a test set is proportional to its number of pairs. Our model and Sultan et al. (2015) are almost identical on all twenty test sets from SemEval 2012–2015, supporting the correctness of our Bayesian implementation.

Following Mohler et al. (2011), for SAS we use RMSE and Pearson's r with gold scores over all answers. These metrics are complementary: correlation is a measure of consistency across students while error measures deviation from individual scores. Our model beats the state-of-the-art text matching model of Mohler et al. (2011) on both metrics.[7]

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

[7] Ramachandran et al. (2015) report better results; however, they evaluate on a much smaller random subset of the test data and use in-domain annotations for model training.

Finally, for ASR, we adopt two metrics widely used in information retrieval: mean average precision (MAP) and mean reciprocal rank (MRR). MAP assesses the quality of the ranking as a whole whereas MRR evaluates only the top-ranked answer sentence. Severyn and Moschitti (2015) report a convolutional neural network model of text similarity which shows top ASR results on the Wang et al. (2007) dataset. Our model outperforms this model on both metrics.

5.1 Adaptation to STS Domains

Ideally, our domain adaptation (DA) should allow the application of large amounts of out-of-domain training data along with few in-domain examples to improve in-domain performance. Given data from n domains, two other alternatives in such scenarios are: (1) to train a single *global* model using all available training examples, and (2) to train n *individual* models, one for each domain, using only in-domain examples. We present results from our DA model and these two baselines on the ten STS datasets discussed in Section 2. We fix the training set size per domain and split each domain into train and test folds randomly.

Models have access to training data from all ten domains (thus nine times more out-of-domain examples than in-domain ones). Each model (global, individual, and adaptive) is trained on relevant annotations and applied to test pairs, and Pearson's r with gold scores is computed for each model on each individual test set. Since performance can vary across different splits, we average over 20 splits of the same train/test ratio per dataset. Finally, we evaluate each model with a weighted sum of average correlations across all test sets, where the weight of a test set is proportional to its number of pairs.

Figure 5 shows how models adapt as the training set grows. The global model clearly falters with larger training sets in comparison to the other two models. On the other hand, the domain-specific model (i.e., the ten individual models) performs poorly when in-domain annotations are scarce. Importantly, the adaptive model performs well across different amounts of available training data.

To gain a deeper understanding of model performance, we examine results in individual domains. A single performance score is computed for every model-domain pair by taking the model's average

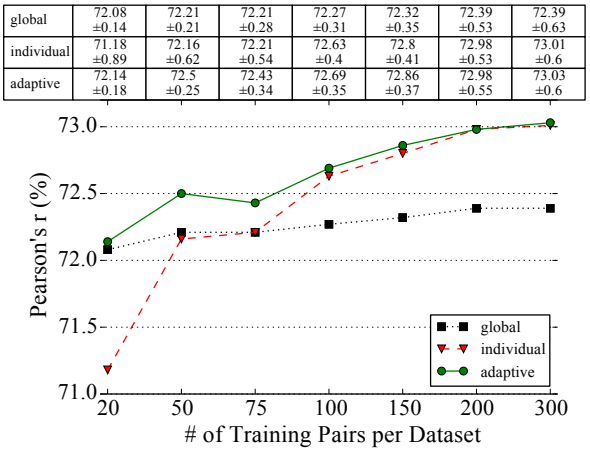


Figure 5: Results of adaptation to STS domains across different amounts of training data. Table shows mean±SD from 20 random train/test splits. While the baselines falter at extremes, the adaptive model shows consistent performance.

Dataset	Glob.	Indiv.	Adapt.
Answers-forums (2015)	.9847	**1**	.9999
Answers-students (2015)	.9850	**1**	.9983
Belief (2015)	**1**	.9915	.9970
Headlines (2015)	.9971	.9998	**1**
Images (2015)	.9992	.9986	**1**
Deft-forum (2014)	**1**	.9775	.9943
OnWN (2014)	.9946	.9990	**1**
Tweet-news (2014)	.9998	.9950	**1**
SMT (2013)	**1**	.9483	.9816
MSRpar-test (2012)	.9615	**1**	.9923
Mean	.9918	.9911	**.9962**
SD	.0122	.0165	.0059

Table 2: Correlation ratios of the three models vs. the best model across STS domains. Best scores are **boldfaced**, worst scores are underlined. The adaptive model has the best (1) overall score, and (2) consistency across domains.

Dataset	Var.	Glob.	Indiv.	Adapt.
SMT	w_1	.577	.214	.195
	w_2	.406	-.034	.134
	r	**.4071**	.3866	**.4071**
MSRpar-test	w_1	.577	1.0	.797
	w_2	.406	-.378	.050
	r	.6178	**.6542**	.6469
Answers-students	w_1	.577	.947	.865
	w_2	.406	.073	.047
	r	.7677	**.7865**	.7844

Table 3: Feature weights and correlations of different models in three extreme scenarios. In each case, the adaptive model learns relative weights that are more similar to those in the best baseline model.

correlation in that domain over all seven training set sizes of Figure 5. We then normalize each score by dividing by the best score in that domain. Each cell in Table 2 shows this score for a model-domain pair. For example, Row 1 shows that—on average—the individual model performs the best (hence a correlation ratio of 1.0) on QA forum answer pairs while the global model performs the worst.

While the adaptive model is not the best in every domain, it has the best worst-case performance across domains. The global model suffers in domains that have unique parameter distributions (e.g., MSRpar-test: a paraphrase dataset). The individual model performs poorly with few training examples and in domains with noisy annotations (e.g., SMT: a machine translation evaluation dataset). The adaptive model is much less affected in such extreme cases. The summary statistics (weighted by dataset size) confirm that it not only stays the closest to the best model on average, but also deviates the least from its mean performance level.

5.1.1 Qualitative Analysis

We further examine the models to understand *why* the adaptive model performs well in different extreme scenarios, i.e., when one of the two baseline models performs worse than the other. Table 3 shows feature weights learned by each model from a split with seventy-five training pairs per domain and how well each model does.

All three domains have very different outcomes for the baseline models. We show weights for the alignment (w_1) and embedding features (w_2). In each domain, (1) the relative weights learned by the two baseline models are very different, and (2) the adaptive model learns relative weights that are closer to those of the best model. In SMT, for example, the predictor weights learned by the adaptive model have a ratio very similar to the global model's and does just as well. On Answers-students, however, it learns weights similar to those of the in-domain model, again approaching best results for the domain.

Now, the labor of cleaning up at the karaoke parlor is realized.	Gold=.52 ΔG=**.1943** ΔI=.2738 ΔA=.2024
Up till now on the location the cleaning work is already completed.	
The Chelsea defender Marcel Desailly has been the latest to speak out.	Gold=.45 ΔG=.2513 ΔI=**.2222** ΔA=.2245
Marcel Desailly, the France captain and Chelsea defender, believes the latter is true.	

Table 4: Sentence pairs from SMT and MSRpar-test with gold similarity scores and model errors (Global, Individual and Adaptive). The adaptive model error is very close to the best model error in each case.

Table 4 shows the effect of this on two specific sentence pairs as examples. The first pair is from SMT; the adaptive model has a much lower error than the individual model on this pair, as it learns a higher relative weight for the embedding feature in this domain (Table 3) via inductive transfer from out-of-domain annotations. The second pair, from MSRpar-test, shows the opposite: in-domain annotations help the adaptive model fix the faulty output of the global model by upweighting the alignment feature and downweighting the embedding feature.

The adaptive model gains from the strengths of both in-domain (higher relevance) and out-of-domain (more training data) annotations, leading to good results even in extreme scenarios (e.g., in domains with unique parameter distributions or noisy annotations).

5.2 Multitask Learning

We now analyze performance of our multitask learning (MTL) model in each of the three tasks: STS, SAS and ASR. Multitask baselines resemble DA's: (1) a global model trained on all available training data (Figure 4), and (2) nineteen task-specific models, each trained on an individual dataset from one of the three tasks (Figure 1). The smallest of these datasets has only 204 pairs (SAS assignment #1); therefore, we use training sets with up to 175 pairs per dataset. Because the MTL model is more complex, we use a stronger regularization for this model (m_{σ_w}=10) while keeping the number of MCMC samples unchanged. As in the DA experiments, we compute average performance over twenty random train/test splits for each training set size.

Figure 6 shows STS results for all models across

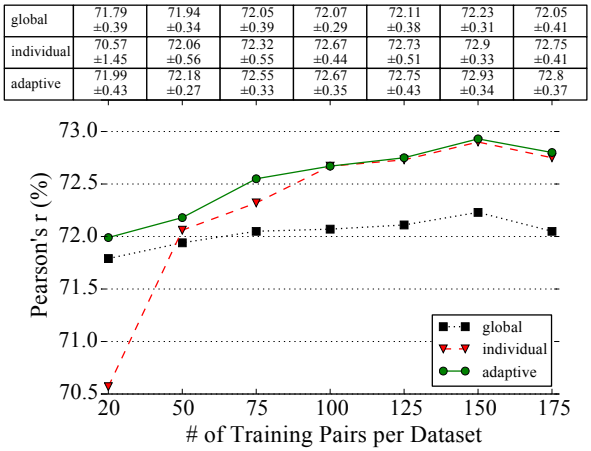


Figure 6: Multitask learning for STS: mean±SD from twenty random train/test splits. The adaptive model consistently performs well while the baselines have different failure modes.

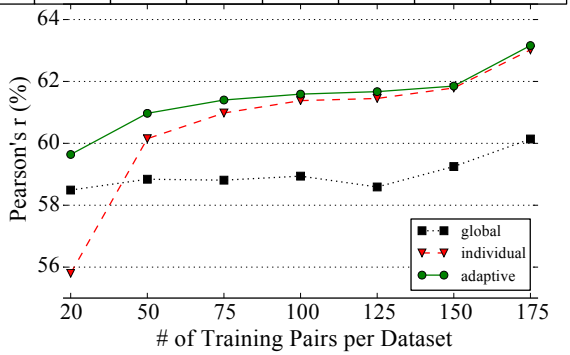
different training set sizes. Like DA, the adaptive model consistently performs well while the global and individual models have different failure modes. However, the individual model does better than in DA: it overtakes the global model with fewer training examples and the differences with the adaptive model are smaller. This suggests that inductive transfer and therefore adaptation is less effective for STS in the MTL setup than in DA. Later in this section, coarse-grained ASR annotations (binary as opposed to real-valued) in MTL may provide an explanation for this.

The performance drop after 150 training pairs is a likely consequence of the random train/test selection process.

For SAS, the adaptive model again has the best overall performance for both correlation and error (Figure 7). The correlation plot is qualitatively similar to the STS plot, but the global model has a much higher RMSE across all training set sizes, indicating a parameter shift across tasks. Importantly, the adaptive model remains unaffected by this shift.

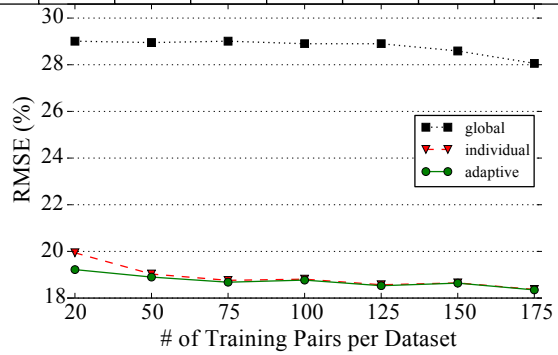
The ASR results in Figure 8 show a different pattern. Contrary to all results thus far, the global model performs the best in this task. The individual model consistently has lower scores, regardless of the amount of training data. Importantly, the adaptive model stays close to the global model even with very few training examples. The ASR datasets are heavily biased towards negative examples; thus, we

global	58.49 ±1.12	58.84 ±0.88	58.81 ±1.18	58.94 ±1.58	58.59 ±2.39	59.25 ±2.79	60.14 ±2.77
individual	55.8 ±4.65	60.15 ±1.86	60.98 ±1.15	61.38 ±2.0	61.45 ±2.21	61.79 ±2.52	63.02 ±2.51
adaptive	59.64 ±1.74	60.97 ±1.51	61.4 ±1.07	61.59 ±1.89	61.67 ±2.3	61.85 ±2.52	63.16 ±2.49



(a) Correlation.

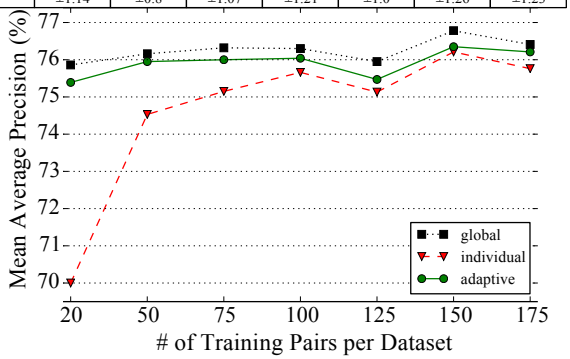
global	29.01 ±0.92	28.95 ±0.66	29.01 ±0.78	28.9 ±0.52	28.9 ±0.68	28.59 ±0.72	28.06 ±0.8
individual	19.94 ±0.88	19.03 ±0.41	18.76 ±0.33	18.81 ±0.45	18.57 ±0.52	18.65 ±0.58	18.37 ±0.84
adaptive	19.22 ±0.32	18.9 ±0.36	18.68 ±0.3	18.77 ±0.44	18.53 ±0.53	18.64 ±0.59	18.35 ±0.83



(b) Error.

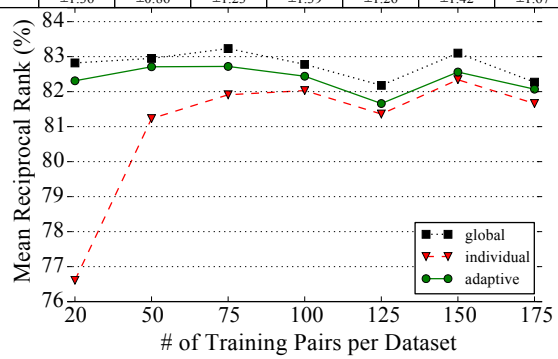
Figure 7: Multitask learning for SAS: mean±SD from 20 random train/test splits. The adaptive model performs the best, and successfully handles domain shift evident from the global model error.

global	75.86 ±0.39	76.16 ±0.8	76.32 ±0.96	76.3 ±1.31	75.95 ±1.22	76.78 ±1.24	76.41 ±1.31
individual	70.0 ±1.45	74.53 ±1.3	75.15 ±1.25	75.66 ±1.27	75.13 ±1.11	76.21 ±1.2	75.76 ±1.17
adaptive	75.39 ±1.14	75.95 ±0.8	76.0 ±1.07	76.04 ±1.21	75.47 ±1.0	76.35 ±1.26	76.21 ±1.23



(a) Mean Average Precision.

global	82.82 ±0.63	82.95 ±0.91	83.23 ±1.15	82.78 ±1.59	82.18 ±1.43	83.1 ±1.3	82.27 ±1.48
individual	76.61 ±4.56	81.23 ±1.64	81.91 ±1.57	82.03 ±1.44	81.36 ±1.37	82.34 ±1.24	81.66 ±1.72
adaptive	82.31 ±1.36	82.71 ±0.86	82.72 ±1.23	82.44 ±1.39	81.66 ±1.26	82.56 ±1.42	82.07 ±1.67



(b) Mean Reciprocal Rank.

Figure 8: Multitask learning for ASR: mean±SD from 20 random train/test splits. Least affected by coarse-grained in-domain annotations, the global model performs the best; the adaptive model stays close across all training set sizes.

use stratified sampling to ensure each ASR training set has balanced examples.

A reason for the global model's strength at ASR may lie in the finer granularity of the real-valued STS and SAS scores compared to binary ASR annotations. If a fine granularity is indeed desirable in training data, as a model that ignores in-domain and out-of-domain distinction, the global model would be affected the least by coarse-grained ASR annotations. To test this hypothesis, we train a linear model on all STS examples from SemEval 2012–2015 and apply it to the ASR test set via a logistic transformation. This model indeed demonstrates better results (MAP=.766, MRR=.839) than our base model trained on ASR annotations (Table 1). This is an unusual scenario where in-domain training examples matter less than out-of-domain ones, hurting domain-specific and adaptive models.

Going back to STS, this finding also offers an explanation of why adaptation might have been less useful in multitask learning than in domain adaptation, as only the former has ASR annotations.

6 Discussion and Related Work

For a variety of short text similarity tasks, domain adaptation improves average performance across different domains, tasks, and training set sizes. Our adaptive model is also by far the least affected by adverse factors such as noisy training data and scarcity or coarse granularity of in-domain examples. This combination of excellent average-case and very reliable worst-case performance makes it the model of choice for new STS domains and applications.

Although STS is a useful task with sparse data, few domain adaptation studies have been reported. Among those is the supervised model of Heilman and Madnani (2013a; 2013b) based on the multilevel model of Daumé III (2007). Gella et al. (2013) report using a two-level stacked regressor, where the second level combines predictions from n level 1 models, each trained on data from a separate domain. Unsupervised models use techniques such as tagging examples with their source datasets (Gella et al., 2013; Severyn et al., 2013) and computing vocabulary similarity between source and target domains (Arora et al., 2015). To the best of our knowledge, ours is the first systematic study of supervised DA and MTL techniques for STS with detailed comparisons with comparable non-adaptive baselines.

7 Conclusions and Future Work

We present hierarchical Bayesian models for supervised domain adaptation and multitask learning of short text similarity models. In our experiments, these models show improved overall performance across different domains and tasks. We intend to explore adaptation to other STS applications and with additional STS features (e.g., word and character n-gram overlap) in future. Unsupervised and semi-supervised domain adaptation techniques that do not assume the availability of in-domain annotations or that learn effective domains splits (Hu et al., 2014) provide another avenue for future research.

Acknowledgments

This material is based in part upon work supported by the NSF under grants EHR/0835393 and EHR/0835381. Boyd-Graber is supported by NSF grants IIS/1320538, IIS/1409287, and NCSE/1422492. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsor.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In *SemEval*.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In **SEM*.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 Task 10: Multilingual semantic textual similarity. In *SemEval*.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 Task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *SemEval*.

Piyush Arora, Chris Hokamp, Jennifer Foster, and Gareth J.F.Jones. 2015. DCU: Using distributional semantics and domain adaptation for the semantic textual similarity SemEval-2015 Task 2. In *SemEval*.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *SemEval*.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count and predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the Association for Computational Linguistics*.

Chris Brockett. 2007. Aligning the RTE 2006 corpus. Technical Report MSR-TR-2007-77, Microsoft Research.

Yee Seng Chan and Hwee Tou Ng. 2008. MAXSIM: A maximum similarity metric for machine translation evaluation. In *Proceedings of the Association for Computational Linguistics*.

Anirban Dasgupta, Ravi Kumar, and Sujith Ravi. 2013. Summarization through submodularity and dispersion. In *Proceedings of the Association for Computational Linguistics*.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the Association for Computational Linguistics*.

Jenny Rose Finkel and Christopher D. Manning. 2009. Hierarchical bayesian domain adaptation. In *Conference of the North American Chapter of the Association for Computational Linguistics*, Morristown, NJ, USA.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Conference of the North American Chapter of the Association for Computational Linguistics*.

Spandana Gella, Bahar Salehi, Marco Lui, Karl Grieser, Paul Cook, and Timothy Baldwin. 2013. UniMelb_NLP-CORE: Integrating predictions from multiple domains and feature sets for estimating semantic textual similarity. In **SEM*.

Christian Hänig, Robert Remus, and Xose de la Puente. 2015. ExB Themis: Extensive feature extraction from word alignments for semantic textual similarity. In *SemEval*.

Michael Heilman and Nitin Madnani. 2013a. ETS: Domain adaptation and stacking for short answer scoring. In *SemEval*.

Michael Heilman and Nitin Madnani. 2013b. HENRY-CORE: Domain adaptation and stacking for text similarity. In *SemEval*.

Yuening Hu, Ke Zhai, Vlad Eidelman, and Jordan Boyd-Graber. 2014. Polylingual tree-based topic models for translation domain adaptation. In *Association for Computational Linguistics*.

Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2011. Better evaluation metrics lead to better machine translation. In *Proceedings of Empirical Methods in Natural Language Processing*.

André Lynum, Partha Pakray, Björn Gambäck, and Sergio Jimenez. 2014. NTNU: Measuring semantic similarity with sublexical feature representations and soft cardinality. In *SemEval*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations Workshop*.

Michael Mohler, Razvan Bunescu, and Rada Mihalcea. 2011. Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the Association for Computational Linguistics*.

Anand Patil, David Huard, and Christopher J. Fonnesbeck. 2010. PyMC: Bayesian stochastic modelling in python. *Journal of Statistical Software*, 35(4).

Lakshmi Ramachandran, Jian Cheng, and Peter Foltz. 2015. Identifying patterns for short answer scoring using graph-based lexico-semantic text matching. In *NAACL-BEA*.

John Salvatier, Thomas V. Wiecki, and Christopher Fonnesbeck. 2015. Probabilistic programming in python using PyMC. *arXiv:1507.08050v1*.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*.

Aliaksei Severyn, Massimo Nicosia, and Alessandro Moschitti. 2013. Learning semantic textual similarity with structural representations. In *Proceedings of the Association for Computational Linguistics*.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2014. Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *TACL*, 2.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2015. DLS@CU: Sentence similarity from word alignment and semantic vector composition. In *SemEval*.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of Empirical Methods in Natural Language Processing*.

Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. 2013. A sentence compression based framework to query-focused multi-document summarization. In *Proceedings of the Association for Computational Linguistics*.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013. Answer extraction as sequence tagging with tree edit distance. In *Conference of the North American Chapter of the Association for Computational Linguistics*.