Domain Adaptation for Commitment Detection in Email

Hosein Azarbonyad
University of Amsterdam
h.azarbonyad@uva.nl

Robert Sim
Microsoft Research AI
rsim@microsoft.com

Ryen W. White
Microsoft Research AI
ryenw@microsoft.com

ABSTRACT

People often make commitments to perform future actions. Detecting commitments made in email (e.g., "I'll send the report by end of day") enables digital assistants to help their users recall promises they have made and assist them in meeting those promises in a timely manner. In this paper, we show that commitments can be reliably extracted from emails when models are trained and evaluated on the same domain (corpus). However, their performance degrades when the evaluation domain differs. This illustrates the domain bias associated with email datasets and a need for more robust and generalizable models for commitment detection. To learn a domain-independent commitment model, we first characterize the differences between domains (email corpora) and then use this characterization to transfer knowledge between them. We investigate the performance of domain adaptation, namely transfer learning, at different granularities: feature-level adaptation and sample-level adaptation. We extend this further using a neural autoencoder trained to learn a domain-independent representation for training samples. We show that transfer learning can help remove domain bias to obtain models with less domain dependence. Overall, our results show that domain differences can have a significant negative impact on the quality of commitment detection models and that transfer learning has enormous potential to address this issue.

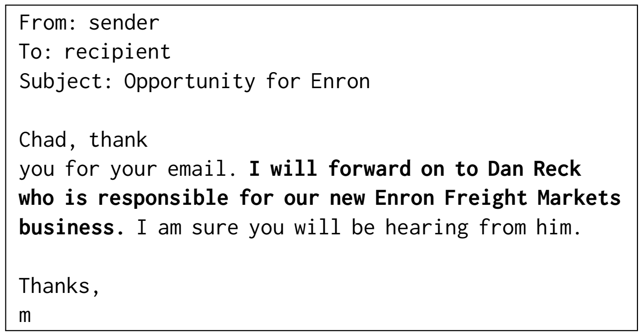
KEYWORDS

Commitment Detection, Email Management, Domain Adaptation

ACM Reference Format:
Hosein Azarbonyad, Robert Sim, and Ryen W. White. 2019. Domain Adaptation for Commitment Detection in Email. In *The Twelfth ACM International Conference on Web Search and Data Mining (WSDM '19), February 11–15, 2019, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3289600.3290984

1 INTRODUCTION

Email is an important communication medium for individuals and organizations. People use email not only as a communication tool, but also as a means to create and manage tasks [6, 13, 36, 44]. When the number of ongoing tasks created via emails increases, people can struggle to manage their tasks and monitor their progress [3, 45]. Automatic task management systems can overcome this problem and help people manage their tasks more efficiently [3, 20]. Commitments such as "I'll send the report by end of day" are one type of task that involve promises made between individuals to complete future actions [12]. Figure 1 shows an example commitment in an email. Such tasks are often hidden in email, and users can struggle to recall and complete them in a timely manner. Detecting commitments



From: sender
To: recipient
Subject: Opportunity for Enron

Chad, thank
you for your email. **I will forward on to Dan Reck who is responsible for our new Enron Freight Markets business.** I am sure you will be hearing from him.

Thanks,
m

Figure 1: A sample email from the Enron email corpus with a commitment sentence highlighted. The commitment detection task is to automatically detect such sentences in email.

automatically enables task management tools and digital assistants to generate reminders and notifications to help users meet their obligations. Despite the potential benefits of automatic commitment detection, work in this important area has been limited to only a handful of studies [12, 15, 26, 27, 31].

Commitment detection is challenging for at least two reasons. First, the commitment detection task itself is inherently difficult, even for humans [26], and particularly difficult when the text is short, with limited context. Second, email models are trained and shipped to users based on potentially biased datasets. For both privacy and practical reasons, email-based models are often trained on public datasets, which are skewed in a variety of ways. For example, Enron [24] and Avocado [37], two commonly-used email datasets for learning email models, belong to two organizations with different focus areas and from different time periods. Terminology, including named entities and technical jargon, can vary greatly across domains and over time. As such, models learned on one dataset may be biased and might not perform well on a different target dataset. Biases will affect all models trained on email corpora. This issue can happen even for general email providers as different email users use different terminology and training a general model that can detect commitments for all users is very challenging. On one hand, it is not realistic to train a separate model for each user as collecting training samples is very costly. On the other, a model trained on emails of a set of users will still be biased. Transfer learning [38] is a domain adaptation method that enables transferring knowledge learned in one domain (source domain) to another domain (target domain). It has been shown that this approach is successful for addressing domain differences in many tasks including text and image classification [40, 46], sentiment analysis [7], and collaborative filtering [30]. Using this approach for transferring knowledge learned from one email collection to another may help achieve more robust and generalizable models for commitment detection.

In this paper, we evaluate the efficacy of transfer learning for commitment detection. To learn commitment detection models, we first collect two datasets with commitments for emails sourced from the Enron and Avocado collections. To do this, we extract sentences from emails and ask trained annotators to assign binary commitment labels to the sentences. We argue that commitment vocabulary is mostly domain independent and that a transfer learning approach

WSDM '19, February 11–15, 2019, Melbourne, VIC, Australia
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-5940-5/19/02…$15.00
https://doi.org/10.1145/3289600.3290984

can help remove domain-specific information from commitment models and capture the core language of commitments.

To learn a domain-independent commitment model, we first try to characterize the differences between domains (email corpora) and then use this characterization to transfer knowledge between them. We investigate the performance of domain adaptation methods working at different granularities: feature-level adaptation and sample-level adaptation. For feature level adaptation, we first learn a mapping between features (n-grams) of source and target domains and use the mapping to transfer the features of the source domain to the target domain and train the classifier using the transferred data. For sample-level adaptation, we use importance sampling (IS) [42] to weight training samples in the source domain based on their similarity to the samples of the target domain. We use the weighted samples in the source domain to train the classifier and apply it to the samples in the target domain. We further combine sample-level and feature-level domain adaptation using a neural autoencoder [5]. The autoencoder is trained to maximize the commitment classification accuracy while minimizing the reconstruction loss. Moreover, to remove domain specific information from representations of samples, a domain classifier is added to the autoencoder. Given a training sample from source/target domain, the domain classifier attempts to predict the domain to which the sample belongs. During training, the accuracy of the domain classifier is minimized. By minimizing domain classifier accuracy, domain-specific information is removed from the samples.

Our main research contributions are as follows:

- Study the impact of domain transfer on commitment detection. We show that the quality of commitment detection degrades significantly as we apply commitment detection models across domains.
- Propose different approaches for characterizing differences between email corpora and show that domain adaptation (specifically, transfer learning) can remove domain-specific bias from commitment detection models.
- Demonstrate through extensive analysis that domain adaptation methods lead to significant gains in the precision and recall of commitment detection models.

The remainder of this paper is organized as follows: we expand on related research in Section 2, and continue with describing the datasets in Section 3. Then we explain the domain adaptation approaches used in this paper in Section 4. The experimental setup and the evaluation approach are described in Section 5. Section 6 describes the results of each of the methods used, and proceeds with a detailed discussion of the validation. We conclude in Section 7 with a brief discussion of implications and possible future directions.

2 RELATED WORK

Prior research in a number of areas applies to the study presented in this paper. This includes work on automatic email management [3, 6, 12, 31, 45] and task progress monitoring [3, 13, 20]. Prior research on commitment detection and domain adaptation is particularly relevant, and we describe it in detail in this section.

2.1 Commitment Detection

The detection of commitments in email has been the subject of several prior studies. Lampert et al. [26, 28] show that the annotation of commitments and requests in email is challenging, even for humans. They devise guidelines for collecting judgments and building datasets for commitment/request classification. The most interesting insight is that when statements are given in context (full email) the annotation task is easier and more accurate. Although the findings of these studies can help design commitment detectors, no automatic commitment classifiers are trained in these studies. De Felice [15] proposes a fine-grained classification of commitments in emails. She further studies which phrases are associated with commitments. As with [26, 28], no automatic commitment detector is created. Automatic commitment classifiers have been developed in prior work. Cohen et al. [12] train classifiers to classify sentences in email into one of the following speech acts: deliver, commit, request, amend, propose, meeting. They represent sentences by TF-IDF weighted vectors over word n-grams. Kalia et al. [23] use more sophisticated features for detecting commitments in email such as named entities, part-of-speech (POS) tags, dependencies, and co-reference resolution. They consider both requests and commitments as commitments. They run experiments on Enron and an instant messaging dataset (Hewlett-Packard's IT incident management chat logs). Corston-Oliver et al. [13] use different kinds of features for detecting tasks in email. They consider commitments as one of the tasks they attempt to extract from email. Lampert et al. [27] also consider both requests and commitments. They use a set of features including message length, the presence of modal verbs, and question words, and train a classifier on a set of manually-labeled emails. Their main conclusion is that only some regions of emails are relevant to commitment detection, and other regions often introduce noise.

All these studies use a single dataset for training models and performing analysis. The datasets used in these studies are also small. The studies do not consider the important challenge of domain bias or domain adaptation. Although [12] perform a limited analysis on the transferability of commitment models, they use datasets from the same domain.

2.2 Domain Adaptation

Domain adaptation is the ability to learn a model from data in a source domain and adapt it to have a good performance on a different target domain [4]. Domain adaptation methods can be grouped into two categories: sample-level adaptation and feature-level adaptation methods [38, 47]. Sample level adaptation methods attempt to remove domain bias by weighting the samples in a way that the difference between the distribution of the weighted samples in source domain and the distribution of samples in target domain is reduced. The most common sample weighting approach is importance sampling, in which source samples are re-weighted based on their similarity to samples on the target side [42]. TrAdaBoost [14] exploits a similar idea, but the re-weighting is performed iteratively in a boosting fashion. However, unlike the importance sampling method, TrAdaBoost needs labeled samples in the target domain. Similar ideas have been used in other studies for sample-level domain adaptation [22, 48].

Feature-level adaptation techniques try to remove the domain bias from features by removing domain specific features or transforming them from source to target domain [25, 38, 47]. Learning a mapping between features of different domains have been studied in machine translation [29, 33]. Here the domains are two different languages. With the success of deep autoencoders for learning unsupervised feature representations, these models have been used for domain adaptation [9, 19, 47]. The main intuition behind these methods is using a set of combined samples from source and target domains to learn a representation that is domain independent. After creating the representation, a classifier can be trained on the de-biased representations. Zhuang et al. [47] integrate the representation learning step of the autoencoders with the classifier learning step. They use an autoencoder which tries to learn a representation that is both domain independent and leads to a good performance in classification of samples in the target domain. In this study, we apply a similar idea, however, we try to further remove the domain bias by introducing a domain classifier. Moreover, Zhuang et al. [47] adapts the model for image classification and it is not straightforward to use this model for detecting commitments.

Recently, adversarial training has been applied to domain adaptation [8, 10, 17, 18, 32]. The main intuition behind these methods is adding a domain classification loss to the task's loss and trying to maximize the domain loss. Maximizing the domain loss ensures that the learned representations do not contain any domain information. We also use a similar approach to adversarial training [18] for learning representations of sentences. We further extend this work by using a sequence-to-sequence autoencoder [43] to the model for learning representations. The main reason for having a sequence-to-sequence autoencoder is that this model is a powerful means for encoding text. Having this autoencoder in the pipeline means that we can learn accurate sentence representations that capture the information present in word sequences.

3 TASK AND DATA

In this section, we introduce the commitment detection task and the dataset we collected for training commitment models.

3.1 Detecting Commitments in Email

As in [13], we define a commitment as any sentence in an email where the sender is promising to do an action which can potentially be added to his/her TODO list or be worthy of a reminder (e.g., sending a document, finishing a report, meeting a colleague). We model commitment detection as a binary classification task.

More precisely, the input in the commitment detection task is a sentence in an email and the output is a binary label indicating whether that sentence constitutes a commitment between the email sender and the recipient. We assume that there is a set of commitment sentences $X = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i is a sentence and y_i is the binary commitment label for x_i assigned by annotators. We use X to train a model and then use it to predict commitment labels for new sentences. We use three different representations to represent sentences in X: 1) bag of word n-grams: for feature-based models (such as logistic regression (LR)) we extract word n-grams (we set $n = 3$) and represent the sentences by the frequency of word n-grams in them. 2) bag of part-of-speech (POS) n-grams: similar to the previous representation but with bag of POS-tagged word n-grams (we set $n = 3$) in sentences instead of bag of word n-grams. We use SPLAT [39] to extract POS tags. 3) sequence of words: for sequence-based models (such as autoencoders), we represent sentences as sequences of words.

3.2 Collected Dataset

We use Enron [24] and Avocado [37] email data to construct commitment datasets. In this section, we briefly describe these datasets, the crowd-sourcing task where third-party annotators labeled sentences for whether they constituted commitments, and the collected commitment datasets.

3.2.1 Crowd-sourcing. The Enron dataset contains emails from 158 users who were mostly senior management of Enron, a natural gas transmission corporation. There are about 200K emails in this dataset. The Avocado dataset contains about 940K emails from employees of a defunct information technology company.

To construct crowd-sourcing tasks, we first extract sentences from emails. From the Enron dataset, we randomly select 61,398 sentences and ask annotators to assign commitment labels to them. A random sample of sentences does not contain a high number of positive samples. Therefore, to collect more positive samples, we train an LR classifier on the collected dataset and use it to extract sentences that are more likely positive. We use bag of word n-grams representations to train the LR model. We first use the trained classifier to assign to each sentence the probability of belonging to the positive class. Then, we perform a weighted random sampling based on these probabilities and select an additional 4,000 sentences to annotate. For extracting sentences from Avocado dataset we use the trained LR model to weight sentences and then we select 13,021 sentences for annotation.

After extracting sentences from Enron and Avocado datasets, we ask crowd-workers to assign commitment labels to the sentences. Each annotation task contains a sentence highlighted in an email and the following question: "Does the highlighted sentence contain a specific action that the sender must complete or is obliged to do? (The action must be on the sender and must not already be complete)." If the answer for this question is yes, we consider it a commitment. Each sentence is labeled by two annotators. If there is a disagreement between two annotators, then the sentence is annotated by a third annotator. A sentence is considered positive if at least two annotators annotate it as positive. The inter-annotator agreement between the annotators based on Krippendorff's α is 0.73, indicating a substantial agreement [1].

3.2.2 Commitment datasets. Table 1 shows the statistics of the created datasets. We only use the annotated sentences as samples and ignore the rest of the email. Enron is a much larger dataset than Avocado. Since most of the samples are picked randomly from the dataset, it contains many more negative samples than positive ones. Conversely, since Avocado samples are picked based on the outputs of a machine-learned classifier, it contains more positive samples. Therefore, this dataset is biased toward the classifier and the distribution of positive and negative samples in this dataset does not reflect the true base rate.

Table 1: The statistics of the commitment datasets.

	Enron	Avocado
# samples	65,398	13,021
# positive samples	3,337	4,484
avg. sentence length	12.1	14.5
median sentence length	10	13

Table 2 shows top 10 most informative Enron n-grams for the positive class extracted based on pointwise mutual information. The Jaccard similarity of this set with the top 10 positive class Avocado features is 43%. (Due to licensing restrictions, the Avocado features may not be published.)

Table 2: The most informative Enron features associated with the positive class.

"i will", "i", "will", "i'll", "let you know", "let you", "call you", "i shall", "we will", "will call"

4 TRANSFER LEARNING FOR DETECTING COMMITMENTS

In this section, we describe the methods used for transferring knowledge between email datasets for commitment detection. Given a set of labeled samples in the source domain S, our goal is to create a model that has a high commitment detection accuracy when it is applied in the target domain T. We use three different approaches for transferring classification knowledge between email domains: feature-level adaptation, sample-level adaptation, and an autoencoder that attempts to leverage both feature- and sample-level adaptation.

4.1 Feature-level adaptation

In this section, we introduce our approaches for adapting feature-level domain information. We use two feature-level adaptation techniques: feature selection and feature mapping.

4.1.1 Feature Selection. The main intuition of feature selection approach for domain adaptation is detecting domain-specific features in source and target domains and removing them from the train and test samples. We assume that we have a set of unlabeled samples, D_S, in the source domain and a set of unlabeled samples, D_T, in the target domain(s). To remove domain specific features from S, we first train a domain classifier using samples in D_S as positives and samples in D_T as negatives. The classifier is trained to discriminate samples of D_S from samples of D_T. Therefore, the most discriminative (informative) features of the classifier are considered domain-specific features. We use logistic regression (LR) for training the classifier and select top K features from the classification model (we set $K = 1000$) and finally replace the selected features with a unique symbol ('DOMAIN-WORD') in training samples from the source domain. We follow a similar procedure to remove domain-specific words from samples of the target domain. After removing domain-specific information from the source and target domain samples, we train a commitment classifier on samples from the source domain and directly use that to predict the commitment labels in the target domain.

4.1.2 Feature Mapping. The feature mapping approach attempts to find equivalent features between source and target domains and transform the features from the source domain to their equivalents in the target domain before training the commitment classifier. Features are considered to be word n-grams ($1 \leq n \leq 3$). The main assumption of the feature mapping method is that for each feature in source domain, there is an equivalent feature in the target domain. Therefore, for each domain specific feature in S, there is a domain specific equivalent in T and our goal is to find these equivalences.

We assume that we have a set of emails in each domain. We extract sentences from the emails and, for each domain, we learn a semantic space in which each feature is represented as a low dimensional neural embedded vector. We use Word2Vec (the Skipgram architecture) [35] to generate the embeddings of features. Finally, we learn a linear transformation between the embedding spaces of source and target domains and use it for transforming features between domains.

Linear mapping-based transformations of embeddings between spaces were previously used for translation [34] and detecting semantic shifts [2]. In this approach, we first pick a set of words as anchors between domains as training samples for learning the mapping. We use stopwords as anchors because they should have the same meaning in both domains and they can serve as fixed points around which features with varying meanings (usages) are located. Using the training samples, the goal is to learn a transformation matrix W^{ST} from domain S to domain T that minimizes the distance between the words and their mapped vectors.

The objective function to minimize is:

$$\underset{W^{ST}}{\operatorname{argmin}} \sum_{w} \left\| W^{ST} V_w^S - V_w^T \right\|^2, \tag{1}$$

The sum is taken over the training features.

We use a standard stopword list with a few additional words added (very frequent words in the corpus) to learn the transformation matrix. We use gradient descent algorithm [41] to optimize the objective function. V_w^S and V_w^T are the embeddings of w in the embedding spaces created for source and target domains, respectively. Using the learned transformation, the mapping of a feature w_S from source domain in target domain is determined as follows:

$$M(w) = \underset{w_T \in F_T}{\operatorname{argmax}} cos(W^{ST} V_w^S, V_{w_T}^T),$$

where *cos* is the cosine similarity and F_T is the set of all features (n-grams) in domain T.

4.2 Sample-level adaptation

As a sample-level adaptation method, we use the importance sampling approach. Importance sampling is a technique in statistics to estimate the parameters of a distribution (target distribution) given samples generated from a different distribution (source distribution) [42]. This technique has been applied for domain adaptation for classification [38]. Assume there are two distributions: $P_S(x,y)$ from which samples in the source domain are generated, and $P_T(x,y)$ from which samples in the target domain are generated. In our setting for the commitment detection task, x is a sentence and y is its corresponding commitment label. The goal is to create a model using labeled samples from S while optimizing the objective (e.g., the classifier loss) for samples in T. The importance sampling approach for classification works as follows. As mentioned, the goal is to find parameters of the classifier that minimize the loss for the samples in T:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathbb{E}_{P_T} l(X,Y,\theta)$$

where θ is parameters of the classifier and $l(X,Y,\theta)$ is the loss of the classification. We can rewrite the above equation as follows:

$$\begin{aligned}
\theta^* &\approx \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} P_T(x_i,y_i) l(x_i,y_i,\theta) \\
&= \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \frac{P_T(x_i,y_i)}{P_S(x_i,y_i)} P_S(x_i,y_i) l(x_i,y_i,\theta) \\
&\approx \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \frac{P_T(x_i)P_T(y_i|x_i)}{P_S(x_i)P_S(y_i|x_i)} P_S(x_i,y_i) l(x_i,y_i,\theta) \\
&\approx \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \frac{P_T(x_i)}{P_S(x_i)} P_S(x_i,y_i) l(x_i,y_i,\theta)
\end{aligned}$$

Note that the assumption in the above derivation is that $P_S(y_i|x_i) = P_Y(y_i|x_i)$, which implies that the conditional probability of classes given the samples is independent of the domain. $\frac{P_T(x_i)}{P_S(x_i)}$ is often referred to as the importance weight. Given the above derivation, we can still train a classifier using the samples from the source domain, however we need to weight the loss on samples by their importance weight. $P_S(x_i)$ and $P_T(x_i)$ are the marginal distributions of samples in the source and target domains respectively. To find θ^*, we need samples in the source side and also an estimation of $P_S(x_i)$ and $P_T(x_i)$.

To estimate $P_S(x_i)$ and $P_T(x_i)$, we use sets of samples from the source and target domains. We consider the target side as positive class and source side as negative class. Then we train a domain classifier to predict for each sample how likely it is to be generated from target domain. We use a simple LR model to train the classifier. For each sample x_i, $P_T(x_i) = p$ and $P_S(x_i) = 1 - p$, where p is the probability that x_i is generated from T assigned by the trained domain classifier to x_i.

4.3 Deep Autoencoder

Deep autoencoders have been been successful in unsupervised feature extraction and representation learning [5]. Since these models are unsupervised, they attempt to model the underlying distribution from which the data is generated. In a transfer learning setting, autoencoders are used to learn a representation for a combined set of samples from source and target domains, thereby aiming to simultaneously represent samples from both domains, and yielding

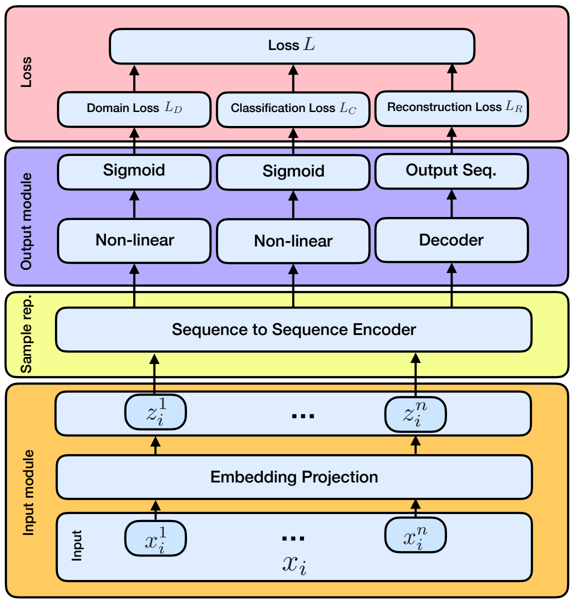


Figure 2: General schema of the proposed neural autoencoder model used for commitment detection.

a model that should be domain independent [9]. After learning the representations model, encodings of samples in the source domain, with their labels, are used for training a classifier.

We first formulate the problem and then describe our proposed model by introducing its general architecture. We then explain the details of various parts of the architecture including input module, sample representation, output module, and loss functions.

The setting is as follows. We are given a set of commitment sentences from the source domain $X_S = \{(x_i^S, y_i^S)\}_{i=1}^N$ and a set of commitment sentences from the target domain $X_T = \{(x_i^T, y_i^T)\}_{i=1}^M$. We train the autoencoder embeddings using unlabeled examples from the source and target domain, and subsequently we apply a set of labeled target examples in addition to the source examples for training the sample representations and output module. Our goal is to use both X_S and X_T to create a domain-independent classifier that has good performance in both domains. Our proposed model takes a candidate sentence from an email and predicts how likely that sentence constitutes a commitment between the email sender and the recipient. To do this, we introduce three different loss functions: reconstruction loss L_R, commitment loss L_C, and domain loss L_D. Reconstruction loss corresponds to how well the learned representations represent the samples. Commitment loss is the main objective which is included to minimize the errors of the commitment classifier. Finally, the domain loss is included to remove the domain bias from samples. In the training process our goal is to maximize the loss of the domain classifier, to avoid capturing domain-specific information during learning sentence representation. Given the described loss functions, the final objective function of the proposed model is as follows:

$$L = \alpha L_R + \beta L_C - \gamma L_D \tag{2}$$

where α, β,and γ control the effect of each loss function on the final loss and they are set based on some preliminary experiments explained in Section 6.4. The details of each loss function are explained in Section 4.3.3. The proposed autoencoder is very similar to a multi-task model, however the main difference is that we minimize the introduced loss functions in a unified framework.

The overall structure of the proposed model is shown in Figure 2. The model contains three primary components:

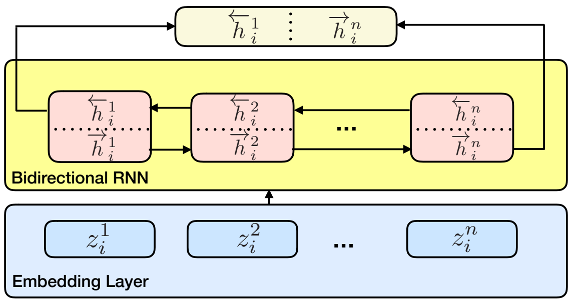


Figure 3: Architecture of the sequence to sequence encoder function in the input representation module.

Input module that provides a set of functions for encoding each input sentence x_i to a sequence of dense vectors $\{z_i^j\}_{j=1}^n$ where $z_i^j \in \mathbb{R}^d$ corresponds to embedding of jth word in the sentence and n is the number of words in the sentence; for simplicity, we assume that d is the dimension of the representation vector of words.

Sample representation that given the outputs of the input module (the sequence of embeddings of words), learns a representation for the input sentence. The output of this module is a vector: $s \in \mathbb{R}^{d1}$, which can be considered as the aggregated representation of the input sentence. We will describe the details of this module in Section 4.3.2.

Output module that captures how likely the input sentence constitutes a commitment based on the representation of the sentence that is provided by the previous module. The details of this unit are explained in Section 4.3.3.

In the following, we explain the input representation, output modules, and loss functions and how these modules are connected.

4.3.1 Input Module. Input module projects an input sentence to a sequence of dense vectors with dimension d using a trainable embedding layer. Each vector in this sequence corresponds to a word in the sentence.

4.3.2 Sample Representation. To represent samples, we use a sequence to sequence recurrent neural network (RNN) as the encoder in our model. The RNN reads the input sequence $Z_i = [z_i^1, z_i^2, ..., z_i^n]$ in the left-to-right direction in the forward pass. It creates a sequence of hidden states, $[\overrightarrow{h}_i^1, \overrightarrow{h}_i^2, ..., \overrightarrow{h}_i^n]$, where $\overrightarrow{h}_i^j = \text{RNN}(z_i^j, \overrightarrow{h}_i^{j-1})$ is a dynamic function for which we can use, for example, an LSTM [21] or a GRU [11]. In this paper, we use an LSTM for learning the representation. The RNN backward pass reads Z_i in the reverse direction, i.e., $\overleftarrow{h}_i^j = \text{RNN}(z_i^j, \overleftarrow{h}_i^{j+1})$, resulting in a sequence of hidden states $[\overleftarrow{h}_i^n, \overleftarrow{h}_i^{k-1}, ..., \overleftarrow{h}_i^1]$. We take the concatenation of the last hidden state of the forward pass and the first hidden state of the backward pass of the RNN, i.e., $\varphi(x_i) = [\overrightarrow{h}_i^k; \overleftarrow{h}_i^1]$, as the final representation for the given data field (see Figure 3).

4.3.3 Output Module. The proposed architecture has three output modules: decoder output, commitment label, and domain label. In this section, we describe each output module.

Decoder output and reconstruction loss. The decoder is an RNN which, given the learned representation for the input sequence (output of the encoder), attempts to generate the input sequence. The goal of the decoder is to estimate the probability $P(o_i|x_i) = P(o_i^1, ..., o_i^{n'}|x_i^1, ..., x_i^n)$, where o_i^t is the output of the decoder at time step t. The decoding starts by reading a special symbol ('GO') at the first time step. Decoding stops by reading another special symbol ('EOS') at the end of each input sentence. Given the outputs at each time step we can determine the decoder's output as follows: $P(o_i|x_i) = \prod_{t=1}^{n'} P(o_i^t|\varphi(x_i), o_i^1, \ldots, o_i^{t-1})$. We train the decoder, in an end-to-end training process in which, given mini-batches of samples $B = \langle x_i, y_i, d_i \rangle$, where y_i and d_i are commitment label and domain label of a sample x_i, we maximize the conditional log-likelihood of a correct output o_i given the input sequence x_i:

$$L_R(x_i) = -\sum_{i=1}^{|B|} log(o_i|x_i) \tag{3}$$

Commitment label and classification loss. The commitment classifier is a feed-forward layer with tanh non-linearity, followed by a sigmoid. It receives the learned representation for the input sentence and predicts the probability of it constituting a commitment:

$$O_C = \tanh(\boldsymbol{W}^{(C)}\varphi(x_i) + \boldsymbol{b}^{(C)}) \qquad \in \mathbb{R}^{d_C}$$
$$\hat{y}_i = \text{sigmoid}(\boldsymbol{w}^T O_C) \qquad \in \mathbb{R},$$

where $\boldsymbol{W}^{(C)} \in \mathbb{R}^{d_C \times d1}$ and $\boldsymbol{b}^{(C)} \in \mathbb{R}^{d_C}$ are a trainable projection matrix and bias respectively, and d_C is the size of projection, and $\boldsymbol{w} \in \mathbb{R}^{d_C}$ is a trainable vector. The commitment classifier is trained in an end-to-end training process. Given mini-batches of data $\langle x_i, y_i, d_i \rangle$, we first predict $\hat{y}$ and calculate the loss using the cross-entropy loss:

$$L_C = -y_i \log\hat{y}_i - (1-y_i)\log(1-\hat{y}_i) \tag{4}$$

Domain label and domain classification loss. As with the commitment classifier, we use a feed-forward layer with tanh non-linearity, followed by a sigmoid, as the output of the domain classifier. The domain classifier predicts the probability of the sentence being generated from the target domain:

$$O_D = \tanh(\boldsymbol{W}^{(D)}\varphi(x_i) + \boldsymbol{b}^{(D)}) \qquad \in \mathbb{R}^{d_D}$$
$$\hat{y}_i = \text{sigmoid}(\boldsymbol{w}^T O_D) \qquad \in \mathbb{R},$$

where $\boldsymbol{W}^{(D)} \in \mathbb{R}^{d_D \times d1}$ and $\boldsymbol{b}^{(D)} \in \mathbb{R}^{d_D}$ are a trainable projection matrix and bias, and d_D is the size of projection, and $\boldsymbol{w} \in \mathbb{R}^{d_D}$ is a trainable vector. The domain classifier is also trained in an end-to-end training process. We again first predict $\hat{y}$ and calculate the loss using the cross-entropy loss:

$$L_D = -d_i \log\hat{y}_i - (1-d_i)\log(1-\hat{y}_i) \tag{5}$$

5 EXPERIMENTAL SETUP

We aim to understand how domain differences can affect the performance of email commitment detection models and how we can use a transfer learning approach to overcome any performance degradation due to these differences. To this end, our main research questions are: (RQ1) Can commitments be reliably detected in emails? (RQ2) How does the performance of commitment models change when they are tested on a different domain than they are trained on? Can we reliably train a model on one domain and use it to detect commitments on a different domain? (RQ3) How can we characterize differences between domains and use this characterization for transferring knowledge between domains? (RQ4) Does the proposed autoencoder help to detect commitments more accurately?

RQ1 is concerned with the quality of automatic commitment detection models. To answer this research question, we use the collected datasets and train and test commitment detection models on the same domain and analyze their performance. RQ2 is concerned with the effect of domain difference on the performance of commitment detection models. To answer RQ2, we train and test commitment models across domains and analyze their performance. To answer RQ3, we try to characterize differences between domains and use this characterization to remove domain-specific bias from commitment models. We evaluate how successful these approaches are in characterizing the differences. RQ4 is concerned with the performance of the proposed autoencoder model in commitment detection and its ability to learn domain independent representations for samples. We report the results of the autoencoder model and compare them to the results of the feature-level and sample-level adaptation approaches.

5.1 Evaluation Metrics

We use standard evaluation metrics for classification such as the area under the receiver operator characteristic curve (AUC), precision, recall, and F1 measure.

5.2 General Setting and Hyperparameters

We set the number of hidden layers of the LSTM (forward and backward) model, d, $d1$, d_C, and d_D to 128. We set the initial learning rate to 10^{-3}. The batch size is set to 128. Training consists of 250K steps. Dropout of the LSTM model is set to 0.2. In training all commitment models, we perform five-fold cross validation and report the average of performance on five folds as the performance of the models.

5.3 Baselines

Since commitment detection is a relatively under-studied task, there are not many baselines to compare our proposed approaches to. Previous work in this area [12, 15, 23, 26, 28] either did not propose a model to automatically detect commitments or used very simple word/POS n-grams based features to train a model. The only differences between the baselines are the classification model and the representation of samples. We also use similar representations (word and POS n-grams) and similar classifiers (such as LR) to create commitment models for this task. In that sense, the LR model is our main baseline. However, our main goal is to show that the domain difference can be very problematic in creating commitment models and previously proposed n-gram based approaches fail to remove domain bias.

For statistical significance testing, we compare our methods to baselines using paired two-tailed t-tests. We set α (the significance level) to 0.05. In Section 6, ▲ and ▼ indicate that the corresponding method performs significantly better or worse than the corresponding baseline, respectively.

6 RESULTS

Following the four research questions described in Section 5, we report the results of our proposed commitment detection methods.

6.1 Commitment Detection Results

To answer RQ1, we use the datasets described in Section 3.2 and train and evaluate commitment classifiers using LR. In this set of experiments, we only focus on the performance of the models trained and tested on same domain. Our goal is to evaluate whether or not commitments can be detected automatically in emails, and whether the LR model can capture the commitment language in emails. Table 3 shows the performance of LR models trained for detecting commitments in Avocado and Enron datasets using word n-gram and POS n-gram representations. The commitment models achieve a reasonable performance. This result indicates that commitments can be reliably detected in emails. There was no significant difference between the performance of models trained on word n-gram and models trained on POS n-grams, and in the remainder of the paper we only report the results based on the word n-gram representation as this representation is more efficient and has lower dimensionality.

6.2 Cross-Domain Results

To answer RQ2, we evaluate the performance of commitment models on a different domain than they are trained on. Again, we use LR for training. Table 4 shows the performance of trained models across domains. The results show that the performance of commitment models

Table 3: Results for LR commitment detection method using word or POS n-grams, trained and tested on same domain.

Dataset	n-grams	Precision	Recall	F1	AUC
Avocado	Word	0.82	0.81	0.81	0.86
	POS	0.82	0.84	0.82	0.86
Enron	Word	0.80	0.77	0.78	0.88
	POS	0.79	0.78	0.78	0.87

degrades when moving across domains in terms of almost all used evaluation measures, and we cannot reliably train a commitment model on one domain and use it to detect commitments on a different domain. For the Avocado⟶Enron case the precision drops more than recall (precision drops from 0.82 to 0.74 and recall drops from 0.81 to 0.78). However, for Enron⟶Avocado case, the opposite is true. The primary reason for higher drops in precision in the Avocado⟶Enron case is that the Avocado dataset contains more positive samples, so the trained model is more inclined towards assigning positive labels to the samples. However, the Enron dataset contains more negative samples. So, for the Avocado⟶Enron case, the false positive rate is high, which leads to lower precision. For Enron⟶Avocado the false negative rate is high (as Enron dataset has more negative samples and the trained classifier is more inclined towards assigning negative labels to samples), which leads to lower recall.

Table 4: Performance of LR method across domains.

Train	Test	Precision	Recall	F1	AUC
Avocado	Avocado	0.82	0.81	0.81	0.86
	Enron	0.77▼	0.69▼	0.73▼	0.67▼
Enron	Enron	0.80	0.77	0.78	0.88
	Avocado	0.74▼	0.78	0.76▼	0.58▼

6.3 Characterizing Inter-Domain Differences

Next, we answer RQ3 by training models that can detect and characterize the differences between email domains and use this characterization for removing domain bias from commitment models before training them. We first design a binary classifier which attempts to identify the source domains for samples. The input of the classifier is a sentence and the output is the domain label for the sentence. We take 5,000 samples from each of Avocado and Enron datasets and represent them using a bag-of-words n-gram representation. Then, we train an LR model to predict domain labels for samples. Figure 4 shows the precision-recall curve of the domain classifier. The classifier achieves an F1 score of 0.85. Table 5 illustrates a sample of features that are strongly associated with the Enron domain. This result indicates that there is a domain bias in the samples and even a simple LR classifier can characterize differences between domains.

Table 5: Most informative unigram features indicating the Enron domain.

“enron”, “gas”, “ena”, “houston”, “ferc”, “eol”, “energy”, “ees”, “counterparty”

We use this characterization to remove domain bias from source and target datasets before training commitment models. Table 6 shows the performance of models that use the characterization of the difference between domains for training domain-independent models. Three observations can be made from the results. First, all transfer

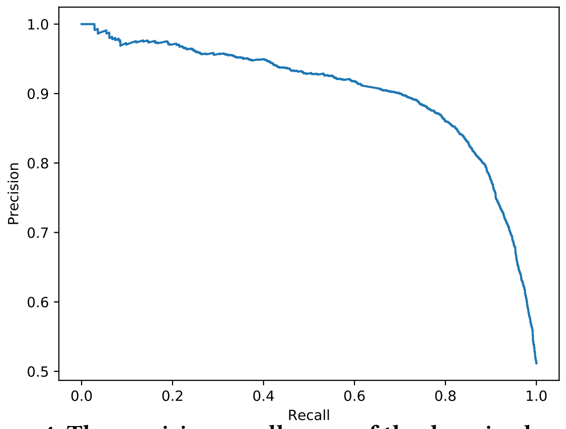


Figure 4: The precision-recall curve of the domain classifier (predicting source domain of the samples) (F1 score = 0.85).

learning models improve the performance of the baseline LR model, indicating that we can remove the domain bias using the transfer learning approach. The improvements of all methods are statistically significant over the LR method for Avocado⟶Enron. However, on Enron⟶Avocado only the importance sampling method achieves significant improvements over the LR method in terms of all metrics. This result indicates that the transfer learning approach is more successful in removing the domain bias from the Avocado dataset. Second, the linear mapping approach has a slightly better performance than the importance sampling method for Avocado⟶Enron. Based on our analysis the quality of the mapping created from Avocado to Enron is higher than the quality of the created mapping from Enron to Avocado. The average cosine similarity of the mapped embedding of words from Avocado embedding (to Enron embedding) to their embedding in Enron space is 0.78. This value for Enron⟶Avocado mapping is 0.69. The better quality of the mapping leads to better transformation from Avocado to Enron and better performance for Avocado⟶Enron. When mapping from Avocado to Enron, the words project to a more meaningful place in the Enron embedding space. Third, the importance sampling method achieves significant improvements for both Avocado⟶Enron and Enron⟶Avocado cases. This indicates that adaptation at the sample level is more effective for the commitment detection task. We use importance sampling as a baseline in Section 6.4 and compare its performance to the performance of the proposed deep autoencoder.

Table 6: Performance of different domain adaptation methods for detecting commitments. IS: Importance Sampling, LM: Linear Mapping, FS: Feature Selection.

Train	Test	Method	Precision	Recall	F1	AUC
Avocado	Enron	LR	0.77	0.69	0.73	0.67
		IS	0.81▲	0.75▲	0.77▲	0.74▲
		LM	0.83▲	0.76▲	0.79▲	0.75▲
		FS	0.80▲	0.73▲	0.76▲	0.73▲
Enron	Avocado	LR	0.74	0.78	0.76	0.58
		IS	0.78▲	0.85▲	0.81▲	0.71▲
		LM	0.75	0.81▲	0.77	0.64▲
		FS	0.74	0.80▲	0.76	0.62

6.4 Transfer Learning Results

To answer RQ4, we evaluate the performance of the proposed autoencoder model in the commitment detection task and compare its performance to that of the importance sampling method. As an additional baseline, we use an LR model trained on a combination of samples in both the Avocado and Enron datasets. Table 7 shows the performance of the autoencoder model with different loss functions. The training of the autoencoder that uses all loss functions is done on the combination of both Avocado and Enron samples with their labels. We again perform five-fold cross validation on the test set. At each step, we use four folds in target side in addition to all samples on source side for training the model and evaluate the trained model on the fifth fold. For training AE_R and AE_{R+D} we train the models using unlabeled samples in the source and target sides. Then, we use the trained model to represent labeled samples from the source side. Finally, we train an LR model using these representations and evaluate its performance on the target side. Note that both AE_R and AE_{R+D} do not use labeled samples in the target domain. Nevertheless, in many cases they outperform models which use labeled samples in the target domain. As expected, adding labeled samples in the target domain to the training set (AE_{All}) improves the performance of the model.

The results show that each introduced loss function contributes to the performance of the autoencoder, and using all loss functions achieves the best performance. Based on some preliminary experiments we set $\alpha = 0.1$, $\beta = 0.6$, and $\gamma = 0.1$ in Equation 2. This indicates that the commitment loss has more effect on the performance of the model, while other losses have the same contribution. The proposed model outperforms both IS and the LR model trained on a combination of the samples from source and target side. This result shows the ability of the proposed autoencoder model to remove the domain bias from data and achieve a robust model.

To observe the effect of the size of training set on the performance of the proposed autoencoder model, we design an experiment in which we vary the number of samples in the target side and measure the performance of the model. Figure 5 shows the results of this experiment. The results show that adding more samples in the target side boosts the performance of the model on both datasets. With about 50% of the samples, the model already achieves a good performance. For the Enron case, after having 50% of the data, adding more samples does not affect the performance of the model significantly. However, for Avocado this is not the case. The main reason is that Enron is a significantly larger dataset and with only 50% of the data the model can already generalize, while Avocado is smaller and adding more samples helps learn a better model.

Table 7: Performance of the proposed autoencoder for domain adaptation for detecting commitments. IS: Importance Sampling, AE_R: Autoencoder with only reconstruction loss, AE_{R+D}: Autoencoder with reconstruction and domain losses, AE_{All}: Autoencoder with reconstruction, domain, and classification losses.

Train	Test	Method	Precision	Recall	F1	AUC
Avocado	Enron	IS	0.81	0.75	0.77	0.74
		LR	0.80	0.79	0.79	0.78
		AE_R	0.80	0.78▲	0.79	0.76
		AE_{R+D}	0.81	0.79▲	0.80▲	0.77▲
		AE_{All}	0.82	0.81▲	0.81▲	0.79▲
Enron	Avocado	IS	0.78	0.85	0.81	0.71
		LR	0.80	0.82	0.81	0.70
		AE_R	0.77	0.82	0.79	0.68
		AE_{R+D}	0.77	0.84	0.80	0.69
		AE_{All}	0.79▲	0.87▲	0.83▲	0.72

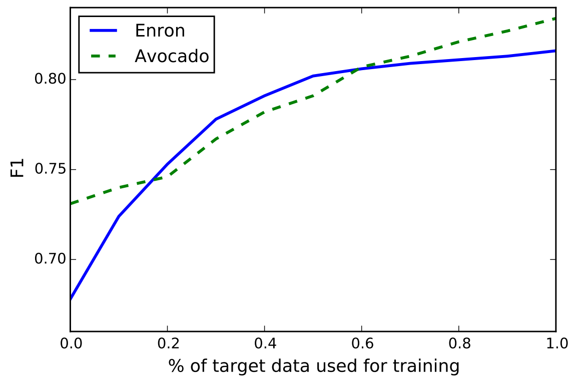


Figure 5: The effect of size of training samples in target domain on the performance of the proposed autoencoder.

7 DISCUSSION AND IMPLICATIONS

We have shown that there is a significant degradation in the performance of commitment detection models across domains (email corpora), that the differences can be accurately characterized, and that domain adaptation, specifically transfer learning, can help ameliorate domain biases and yield performance improvements.

We evaluated the performance of the methods on two publicly available email datasets: Avocado and Enron. The use of public data sets improves the replicability of our study and is sufficient to demonstrate the value of domain adaptation for commitment detection. In internal testing, we conducted further experiments where we evaluated these methods on proprietary email collections sourced within our organization (training on Avocado+Enron datasets combined) and found similar results to those reported thus far in this paper. Nondisclosure and privacy policies do not permit us to discuss the data or the results in detail, but we believe that it is still valuable to report that the methods do scale to test domains beyond Avocado and Enron.

The created commitment dataset based on Enron collection has much fewer positive samples. To have more positive samples, we biased the Avocado dataset. Generating the Avocado dataset in the same way as the Enron dataset would allow us to better understand the models and their transferability, however, creating such a large dataset is costly and beyond the scope of this study.

The designed methods differ a lot in their efficiency. The deep autoencoder model required about 12 hours to train on average and about 50 milliseconds to extract a representation for a test sample, while on a same machine the importance sampling methods take about 4 minutes to train and several microseconds to test on average. Although in training phase the autoencoder is quite slow, it is more efficient during testing.

We attempted to explore the effectiveness of models that work in different granularities and can be applied easily to the task. The demonstrated viability of transfer learning is promising. The effectiveness of other domain adaptation methods such as [14], and alternatives such as multi-task learning models [32], needs to be explored. Beyond commitments, the value of transfer learning for other detection tasks in email, e.g., extracting requests and detecting task completion [16, 23], should be investigated, as should the applicability of these methods to media beyond email (e.g., SMS, instant messaging, meeting transcripts). In this paper, we applied our models to transfer knowledge between two domains. However, the proposed approaches are easily extensible to consider more than two domains.

The transfer learning methods used in this study (except for importance sampling, feature selection, and linear mapping), require labeled samples in the target domain as well as the source domain. In applying these methods, additional data collection efforts would be required to obtain these additional labels and the cost of that should be factored into decisions regarding their application. For reference, the autoencoder model, which uses all loss functions, achieves a good performance with just a few thousand labeled samples in the target domain.

Future work involves using the domain adaptation methods in practice to train more domain independent models and deploying them as skills and add-ins inside digital assistants and task management tools, respectively. Deployment would enable the computation of online performance measures based on implicit and explicit feedback from users making the commitments (rather than offline labeling from third-party judges).

REFERENCES

[1] Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics* 34, 4 (2008), 555–596.

[2] Hosein Azarbonyad, Mostafa Dehghani, Kaspar Beelen, Alexandra Arkut, Maarten Marx, and Jaap Kamps. 2017. Words Are Malleable: Computing Semantic Shifts in Political and Media Discourse. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management.* 1509–1518.

[3] Victoria Bellotti, Nicolas Ducheneaut, Mark Howard, and Ian Smith. 2003. Taking email to task: the design and evaluation of a task management centered email tool. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* 345–352.

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A Theory of Learning from Different Domains. *Machine Learning* 79, 1-2 (2010), 151–175.

[5] Yoshua Bengio et al. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning* 2, 1 (2009), 1–127.

[6] Paul N Bennett and Jaime Carbonell. 2005. Detecting action-items in e-mail. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval.* 585–586.

[7] John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics.* 440–447.

[8] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. In *Proceedings of Advances in Neural Information Processing Systems.* 343–351.

[9] Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. *arXiv preprint arXiv:1206.4683* (2012).

[10] Xilun Chen and Claire Cardie. 2018. Multinomial Adversarial Networks for Multi-Domain Text Classification. *arXiv preprint arXiv:1802.05694* (2018).

[11] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[12] William W Cohen, Vitor R Carvalho, and Tom M Mitchell. 2004. Learning to Classify Email into"Speech Acts". In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing.*

[13] Simon Corston-Oliver, Eric Ringger, Michael Gamon, and Richard Campbell. 2004. Task-focused summarization of email. *Text Summarization Branches Out* (2004).

[14] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. 2007. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning.* 193–200.

[15] Rachele De Felice. 2013. A corpus-based classification of commitments in Business English. In *Yearbook of Corpus Linguistics and Pragmatics 2013.* 153–171.

[16] Michael Gamon, Saliha Azzam, Yizheng Cai, Nicholas Caldwell, and Ye-Yi Wang. 2013. Automatic Task Extraction and Calendar Entry. (2013). US Patent App. 13/170,660.

[17] Yaroslav Ganin and Victor Lempitsky. 2014. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495* (2014).

[18] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research* 17, 1 (2016), 2096–2030.

[19] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11).* 513–520.

[20] Jacek Gwizdka. 2002. Reinventing the inbox: supporting the management of pending tasks in email. In *CHI'02 Extended Abstracts on Human Factors in Computing Systems.* 550–551.

[21] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[22] Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th annual meeting of the association of computational linguistics.* 264–271.

[23] Anup K Kalia, Hamid R Motahari-Nezhad, Claudio Bartolini, and Munindar P Singh. 2013. Monitoring commitments in people-driven service engagements. In *Services Computing (SCC), 2013 IEEE International Conference on.* 160–167.

[24] Bryan Klimt and Yiming Yang. 2004. The enron corpus: A new dataset for email classification research. In *Proceedings of European Conference on Machine Learning.* 217–226.

[25] Wouter M. Kouw, Laurens J.P. van der Maaten, Jesse H. Krijthe, and Marco Loog. 2016. Feature-Level Domain Adaptation. *Journal of Machine Learning Research* 17, 171 (2016), 1–32.

[26] Andrew Lampert, Robert Dale, and Cecile Paris. 2008. Requests and Commitments in Email are More Complex Than You Think: Eight Reasons to be Cautious. In *Proceedings of the Australasian Language Technology Association Workshop 2008.* 64–72.

[27] Andrew Lampert, Robert Dale, and Cecile Paris. 2010. Detecting emails containing requests for action. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics.* 984–992.

[28] Andrew Lampert, Cécile Paris, Robert Dale, et al. 2007. Can requests-for-action and commitments-to-act be reliably identified in email messages. In *Proceedings of the 12th Australasian Document Computing Symposium.* 48–55.

[29] Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Herve Jegou. 2018. Word translation without parallel data. In *Proceedings of International Conference on Learning Representations.*

[30] Bin Li, Qiang Yang, and Xiangyang Xue. 2009. Transfer learning for collaborative filtering via a rating-matrix generative model. In *Proceedings of the 26th annual International Conference on Machine Learning.* 617–624.

[31] Chu-Cheng Lin, Dongyeop Kang, Michael Gamon, Madian Khabsa, Ahmed Hassan Awadallah, and Patrick Pantel. 2017. Actionable Email Intent Modeling with Reparametrized RNNs. *arXiv preprint arXiv:1712.09185* (2017).

[32] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-task Learning for Text Classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics.* 1–10.

[33] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168* (2013).

[34] Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting Similarities among Languages for Machine Translation. In *Proceedings of International Conference on Learning Representations.*

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of Advances in Neural Information Processing Systems.* 3111–3119.

[36] Hamid R Motahari Nezhad, Kalpa Gunaratna, and Juan Cappi. 2017. eassistant: Cognitive assistance for identification and auto-triage of actionable conversations. In *Proceedings of the 26th International Conference on World Wide Web Companion.* 89–98.

[37] Douglas Oard, William Webber, David Kirsch, and Sergey Golitsynskiy. 2015. Avocado research email collection. *Linguistic Data Consortium* (2015).

[38] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[39] Chris Quirk, Pallavi Choudhury, Jianfeng Gao, Hisami Suzuki, Kristina Toutanova, Michael Gamon, Wen-tau Yih, Lucy Vanderwende, and Colin Cherry. 2012. MSR SPLAT, a language analysis toolkit. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstration Session.* 21–24.

[40] Rajat Raina, Andrew Y Ng, and Daphne Koller. 2006. Constructing informative priors using transfer learning. In *Proceedings of the 23rd International Conference on Machine learning.* 713–720.

[41] Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The annals of mathematical statistics* (1951), 400–407.

[42] Hidetoshi Shimodaira. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference* 90, 2 (2000), 227–244.

[43] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of Advances in neural information processing systems.* 3104–3112.

[44] Gokhan Tur, Andreas Stolcke, Lynn Voss, Stanley Peters, Dilek Hakkani-Tur, John Dowding, Benoit Favre, Raquel Fernández, Matthew Frampton, Mike Frandsen, et al. 2010. The CALO meeting assistant system. *IEEE Transactions on Audio, Speech, and Language Processing* 18, 6 (2010), 1601–1611.

[45] Steve Whittaker and Candace Sidner. 1996. Email overload: exploring personal information management of email. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* 276–283.

[46] Pengcheng Wu and Thomas G Dietterich. 2004. Improving SVM accuracy by training on auxiliary data sources. In *Proceedings of the twenty-first International Conference on Machine learning.* 110.

[47] Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. 2015. Supervised Representation Learning: Transfer Learning with Deep Autoencoders. In *Proceedings of the 24th International Conference on Artificial Intelligence.* 4119–4125.

[48] Fuzhen Zhuang, Ping Luo, Hui Xiong, Yuhong Xiong, Qing He, and Zhongzhi Shi. 2010. Cross-domain learning from multiple sources: A consensus regularization perspective. *IEEE Transactions on Knowledge and Data Engineering* 22, 12 (2010), 1664–1678.