Issue Status and Dialogue Act Prediction for Asynchronous Online Social Media User Engagement

Author 1
Address 1
Address 2
place, country
email@abc.com

Author 1
Address 1
Address 2
place, country
email@abc.com

ABSTRACT

In this paper, we present a multitask learning method for predicting the resolution status of the issues expressed in social media conversations among customer-care agents and social media users, along with the nature of dialogues of those conversations. Our method extends beyond social media conversation analysis, and is naturally applicable to general *multiple* sequence labeling tasks where each example sequence has multiple label sequences. Our method learns multiple models, one model for each task, i.e., issue status prediction task and dialogue act prediction task. Each model computes the joint probability of both label sequences (dialogue act and issues status) given the example sequence, i.e., conversation among customers and agents. Such multiple models are learned *simultaneously* by facilitating the learning transfer among models through *explicit parameter sharing*. We experiment the proposed method on real social media conversations dataset collected from Twitter as well as on a publicly available NLP dataset, and show that our method outperforms the state-of-the-art method. In addition, we illustrate how the issue status and dialogue act prediction tasks can be an integral part of socially aware customer care engagement system.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Keywords

multitask learning, multilabel learning, label dependency

1. INTRODUCTION

Over the last few years, explosive growth of users' activity on popular social media channels such as Twitter, Facebook, etc., has given various product/service providers opportunity to identify and engage with their customer-base in a proactive manner. A recent survey of 320 active social media users [7] reveals that three quarters (74%) of users choose brands/services based upon other customers' experience shared online whereas 38% of users engage actively with brands' customer service. Evidently, the social media presence of active users provides opportunities to transform traditional customer relationship management(*CRM*) systems into their *socially aware* counterparts, also termed as *social-CRM*. Through these social-CRM systems, customer service agents can engage in online conversations with customers via social media channels, address their concerns, know their views about a certain product, or even market new products. In a social-CRM system, customer-agent engagement is of asynchronous nature and often takes place among multiple customers and multiple agents (many-to-many), which makes it hard for these systems to keep track of the progress of these conversations, e.g., for engaging social media users for customer care activities for their issue resolution and service/product feedback analysis. Since one agent is usually involved with several customers at different time instances, the agent needs to come back to only those issues which system has identified to be requiring further attention. In addition to identifying these *unresolved* issues, the Social-CRM system should also keep track of the *resolved* issues along with their solutions in order to provide this information to other agents for more effective issue resolution in future. Given the large number of customers and their data, it is important to automate the process of detecting resolved and unresolved issues, based upon the social media conversation between agent and the user. We focus on the problem of automatically predicting the status of an issue, referred as issue status prediction problem.

Often the nature of dialogues (also referred as dialogue act [29]) between agent and user can indicate the current status of the issue. For example, the nature of dialogue can be a simple greeting message, an acknowledgement, or a complaint followed by acknowledgement and answer that accompany the technically involved solution present in the conversation. In addition, nature of dialogues also helps CRM solutions to determine the effectiveness of agents and of her conversational style (see Section 6). Therefore, we also focus on this problem of predicting dialogue acts of the conversations between a customer and agent(s), referred as dialogue act prediction problem. For both of these problems i.e. issues status prediction and dialogue act prediction, there exist a correlation between them. An example of a conversation showing the correlation between current issue status and its dialogues is given in Table 1. From this ta-

KDD '14 New York, USA

Table 1: Example of a conversation between a customer and customer-care agent. Each conversation text (column 3) has two labels associated with it (column 1 and 2).

Dialogue Act	Issue Status	Text
COMPLAINT	OPEN	this mobile is making my life hard
REQEST	OPEN	@user_x is there anything we in the social media team can assist you with?
COMPLAINT	OPEN	so i can't make or receive calls & i can't send texts or receive them.
REQUEST	OPEN	@user_x what is your zip code ?
ANNOUNCEMENT	OPEN	we can check for outages for you.
ANSWER	SOLVED	@user_x your experiencing tower outage.
ANSWER	SOLVED	it is estimated to be cleared by the 21st of february.

ble, we see that customers typically have complaint in their tone while describing their issue with a certain product. Incidentally, issues status `open` tends to have correlation with dialogue class `complaint`. Similarly, correlations may exist between dialogue act `answer` and issue status `closed`. A formal list of issue status types and dialogue act types is given in Tables 3 and 2. Such correlations among labels are not specific to customer-care domain but they have also been exhibited in other domains such as natural language processing (NLP) where words in a sentence can be labeled with their Part of Speech (POS) tags as well as NP chunks [32]. Since the underlying sentence of both POS tagging and NP chunking problems is same, both problems can be cast as one single problem —multilabel sequence labeling problem— where an example sequence (i.e. sentence) has multiple labels sequences (i.e., POS tags and NP chunks).

Similar to multiple labels sequencing problem in NLP domain, both issue-status and dialogue act prediction problems can be formulated as sequence labeling problems because of the inherent sequential structure in the conversation. Supervised sequence classification methods such as conditional random fields [20] provide a natural framework to solve these problems [19, 25, 8]. One can build two separate classification models, one for each problem, either assuming that these two problems are unrelated, or ignoring any relatedness structure. However, as we show in the experiment section, it is more reasonable to exploit the correlation among the issue status and dialogue act label sequences. If we define a task as learning from pairs of example sequence and its corresponding label sequence, then we can cast learning multiple label sequences as *multitask* sequence labeling learning problem[31].

In machine learning, Multitask Learning(MTL) provides a mechanism to learn various related tasks simultaneously such that learning from one task can benefit other task and vice-versa. Often in MTL, multiple tasks are learned together by sharing their parameters explicitly either in a Bayesian way [21, 34, 33] or in a non-Bayesian way [6, 24, 16]. In MTL, most of the work has focused on classification or regression problems, with very little work on sequence labeling problem. In addition, most of the MTL methods are not especially designed for *our* multitask setting, i.e., an example sequence has multiple label sequences. Any method designed especially for multiple label sequences setting should exploit the dependencies among labels. Furthermore, to the best of our knowledge, we are not aware of any MTL method that focuses on dialogue-act/issue-status classification tasks. Most of the dialogue act classification methods that treat a conversation as a sequence and employ structured prediction methods such as [29, 3, 11, 22, 17, 25, 8, 19], focus on the feature engineering aspect of the problem rather than the underlying algorithmic framework. One closest approach to ours is factorial CRF [31], where authors exploits the label dependencies among multiple tasks *implicitly*. In contrast, our proposed method exploits the correlations present in multiple label sequences *explicitly* that not only improves upon the factorial CRF but also leads to a flexible framework for multitask sequence learning.

In this work, we extend the MTL setting to the general sequence labeling problem with multiple label sequences, and propose a novel method for learning from multiple sequence labeling tasks simultaneously. Our method —based on conditional random fields (CRFs)— not only exploits label dependencies but also learns multiple tasks simultaneously by *explicitly* sharing parameters. In our method, we learn one model for each task [1]. Each task has two factors (as opposed to one factor in CRFs), one factor corresponding to *all* labels (we call it *label dependency factor*), and other factor corresponding to task-specific *primary* label (we call it *task-specific factor*). Since the factor corresponding to *all* labels appear in all tasks, we facilitate the learning transfer among tasks by keeping the parameters corresponding to this factor same across all tasks. We show through a variety of experiments on datasets from two different domains that such a model outperforms the state-of-the-art [31] method. Note that learning from multiple labels is typically done in two ways: (1) build one single model that incorporates factors of all label sequences and example sequence, i.e., complete dependency and no independent learning (2) build multiple CRF-like *independent* models with no learning transfer among models, i.e., complete independent learning, and no dependency among labels. the proposed method is a middle ground between these two extremes, and provides the best of both worlds. Because of a task-specific factor, it allows model to learn independently, and at the same time, because of label-dependency factor, it allows learning to be transferred among all tasks.

In addition to the parameter sharing framework, we also propose a variation where label dependency factor is further broken into two parts, one that contain information specific to the task and other information common to all tasks. This variation allows one to control the amount of transfer among multiple tasks. Experimental results of this variation show further improvements.

Our contributions: (1) we propose a novel method for se-

[1]A task definition is expanded to include all labels. A task, for our method, is defined as learning from tuples of example sequence and its label sequences. Each task has one *primary* label sequence, and other label sequences are considered *secondary*.

quence labeling problem for multiple labels sequences. (2) We show the application of our method on real customer conversation dataset from social-CRM domain and standard CoNLL dataset[32], and show improvements upto 6% over the baseline [31]. (3) We propose a variation of this model that adds flexibility in terms of allowing one to control the amount of transfer among tasks. (4) The proposed method is naturally applicable to semi-supervised setting. It provides multiple models that can be used in co-training to incorporate unlabeled examples.

2. BACKGROUND AND PROBLEM DESCRIPTION

We extend the mathematical framework of conditional random field (CRF) [20] to support sequence labeling with multiple labels. Before describing our approach in detail, we first setup mathematical notations and summarize the CRF model. CRFs are undirected graphical models that model the conditional probability of a label sequence given an observed example sequence. Let $\mathcal{G}$ be an undirected graphical model over random variables $\mathbf{x}$ and $\mathbf{y}$ which represents sequence of random variables, i.e. $\mathbf{x} = (x_1, x_2, \dots x_T)$ is the sequence of observed entities (e.g. words in a sentence) that we want to label with $\mathbf{y} = (y_1, y_2, \dots y_T)$. $(\mathbf{x}, \mathbf{y})$ together constitute an example-label pair. In the undirected graph $\mathcal{G}$, let $\mathcal{C} = \{C_1, C_2 \dots\}$ be the set of cliques contained in the graph $\mathcal{G}$ where $C_i = \{\mathbf{y}_c, \mathbf{x}_c\}$, $\mathbf{y}_c \subset \mathbf{y}$ and $\mathbf{x}_c \subset \mathbf{x}$. Given such a graph defined on example-label pair, the conditional probability of a labeled sequence $\mathbf{y}$ given an observed example sequence $\mathbf{x}$ can be written as:

$$p(\mathbf{y}|\mathbf{x}, \theta) = \frac{1}{Z(\mathbf{x})} \prod_{c \in \mathcal{C}} \Phi(\mathbf{y}_c, \mathbf{x}_c|\theta), \tag{1}$$

where Φ is the potential function defined over a clique, and is the function of all random variables in that clique. For example, in a specific case of linear chain CRF, these potential functions are defined over cliques (x_t, y_{t-1}, y_t). Here θ is the parameter, which we include in the potential function to denote that potential functions are parametric functions. $Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{c \in \mathcal{C}} \Phi(\mathbf{y}_c, \mathbf{x}_c|\theta)$ is the partition function which makes sure that the potential functions are normalized, and (1) can be interpreted as probabilities. Usually the potential functions in (1) factorize over the features of the clique and are defined using the exponential function of the form $\Phi(\mathbf{y}_c, \mathbf{x}_c|\theta) = \exp\left(\sum_k \theta_k f_k(\mathbf{y}_c, \mathbf{x}_c)\right)$. Here f_k are the features functions, and θ_k are parameters. The feature functions f_k can be defined arbitrarily which is one of the primary advantages of CRFs. For example, *part of speech* (POS) tagging problem can be modeled as linear chain CRF, where feature functions can be defined over words, their characteristics, and their POS labels. In such a linear chain, indexed with t, a clique is defined for each $(y_{t-1}, y_t, x_t))$ combination. For such a clique, one feature function could be a binary test: $f_k(y_{t-1}, y_t, x_t)$ has value 1 if and only if y_{t-1} has the label ARTICLE, y_t has the label NOUN, and the word x_t begins with a capital letter. A pictorial representation of CRF is given in Figure 1.

2.1 Multitask Sequence Labeling

In multitask sequence labeling problem, we are given multiple label sequences for each example sequence, i.e., in addition to $\mathbf{y} = (y_1, y_2, \dots y_T)$ (as defined for CRFs), we have $\mathbf{z} = (z_1, z_2, \dots z_T)$ as another set of label sequence for $\mathbf{x}$. For simplicity, we only consider two types of label sequences, however, it is straightforward to extend our approach to more than two labeling sequences (see Definition 1). Thus our training examples for the entire task become triplets of $(\mathbf{x}, \mathbf{y}, \mathbf{z})$. We have n such training examples, i.e., $\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}_{i=1}^n$. Therefore, the multiple sequence labeling problem can be formalized as modeling conditional density $p(\mathbf{y}, \mathbf{z}|\mathbf{x})$.

3. OUR APPROACH

In this section, we first describe a basic approach. When modeling the joint probability in multiple label setting, it is a standard practice to build just one model considering all possible factors (or cliques) from example sequence and label sequence [31, 23]. Although proven to be better that building two separate standard CRFs, this approach has many drawbacks (see experiments). One of them is the ability to model the tasks independently. The standard CRF though provides this capability, they do not include the effect of other labels; while other models, e.g., [31, 23] do not provide this capability at all – they only build one single model. In our basic but novel approach, we begin by providing a middle ground between these two extremes (i.e. one single fully dependent model and two fully independent models), where both tasks are modeled independently but at the same time, one task draws benefit from other task through label dependencies.

We model $p(\mathbf{y}, \mathbf{z}|\mathbf{x})$ by considering two types of cliques (and potential function defined on those cliques). The first type of clique, similar to the one in linear chain CRF, consists of adjacent labels in *one* (of any) sequence $\mathbf{y}$, i.e., (y_{t-1}, y_t) along with current x_t i.e. (y_{t-1}, y_t, x_t), and the second type of clique consists of the pair of labels (y_t, z_t) along with current x_t i.e. (y_t, z_t, x_t). Here the first type of clique provides the independence while the second type of clique provides the benefit from other labels. As we shall see later, such a model provides *better discriminating power* than the models that consider all types of cliques [31, 23]. Given such two types of cliques and the potential functions defined over them, the conditional probability of both label sequences given the example sequence can be written as:

$$p^y(\mathbf{y}, \mathbf{z}|\mathbf{x}, \theta^y, \psi^y) = \frac{1}{U^y(\mathbf{x})} \prod_{t=1}^{T} \Big(\underbrace{\Phi(y_{t-1}, y_t, x_t|\theta^y)}_{\text{task(y) factor}} \Big) \Big(\underbrace{\Phi(y_t, z_t, x_t|\psi^y)}_{\substack{\text{label dependency} \\ \text{factor}}} \Big) \tag{2}$$

Similar to CRF, $U^y(\mathbf{x})$ is the normalization factor. Although (2) provides the probability of both the labels, i.e., $(\mathbf{y}, \mathbf{z})$, conditioned on observed data sequence $\mathbf{x}$, there is no clique that depends on adjacent z labels, i.e., z_t, z_{t-1}. Thus though incorporating partial information from other label z, the above model still focuses on the task y. So the above model is only defined for the task y since its primary focus is label y - because of the task y factor. Similarly we can define a model for task z:

$$p^z(\mathbf{y}, \mathbf{z}|\mathbf{x}, \theta^z, \psi^z) = \frac{1}{U^z(\mathbf{x})} \prod_{t=1}^{T} \Big(\underbrace{\Phi(z_{t-1}, z_t, x_t|\theta^z)}_{\text{task(z) factor}} \Big) \Big(\underbrace{\Phi(y_t, z_t, x_t|\psi^z)}_{\substack{\text{label dependency} \\ \text{factor}}} \Big) \tag{3}$$

It is to be noted in the above models that the first clique is the task-specific clique as it only considers the label from one task, while the second clique is the common clique as it takes labels from both tasks. Since the second clique is

common in both models, the first clique (and label) is the model's defining clique (and label), and corresponds to the task that model is built for. Also note that in the above models each type of clique has its own parameters, i.e. task y has its parameters θ^y and ψ^y and the task z has its own parameters θ^z and ψ^z. Such a model where each task has its own set of parameters, we call it UNSHARED model. A pictorial representation of this UNSHARED model is shown in Figure 2. Observe that there are two different models, one for each task. Both models have their own factors (and parameters).

The above models can be optimized (and inferenced) using the standard machinery used in CRF since these models are exactly the same as CRF except an additional clique.

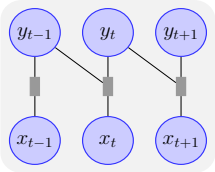

Figure 1: Conditional Random Fields (CRFs)

Below we define the generalized UNSHARED multilabel model, i.e., there can be any number of labels with arbitrary dependencies among them.

DEFINITION 1. *Let* $\mathbf{x}$ *be an observed example sequence with* $\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_k$ *its multiple label sequences. Let* $\mathcal{C}_t$ *be the set of cliques denoting the possible interactions among labels at time* t *(i.e., interaction among labels* $\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_k$*), then, the* UNSHARED *multilabel model is a set of task-specific models where each task-specific model (for task* $\mathbf{y}_l$*) is defined as:*

$$p^{y_l}(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k|\mathbf{x}, \theta^{y_l}, \psi^{y_l}) = \frac{1}{Z(\mathbf{x})}\Big(\prod_{t=1}^{T}\Phi(y_{l,t-1}, y_{l,t}, x_t|\theta^{y_l})\Big)$$
$$\Big(\prod_{t=1}^{T}\prod_{c\in\mathcal{C}_t}\Phi(\mathbf{y}_c, x_t|\psi^{y_l})\Big) \quad (4)$$

3.1 Shared Models

Although more accurate than the existing methods (CRF and factorial CRF) (see experiments), this method does not take advantage of the multitask nature of the problem, as both models have their own separate set of parameters, and there is no learning transfer between these models. We exploit the multitask nature of the problem and facilitate learning transfer by sharing the parameters corresponding to the common clique in both models. Sharing parameters to facilitate learning transfer is a well-known practice in multitask learning [24, 16, 4, 6, 5]. In other words, we make

$$\psi^y = \psi^z = \psi.$$

We call this formulation SHARED model. A pictorial representation of this SHARED model is shown in Figure 2. We emphasize in this figure that there are two *separate* models, with one set of factors that is common to both models. The figure should not be confused for the graphical model for one single model. The parameters corresponding to the common factor are shared between both models, as opposed to UNSHARED model where both models have their own parameters.

Now for the clarity and follow up discussion, we write the formulations (2) and (3) in terms of corresponding feature functions (under SHARED model):

$$p^y(\mathbf{y}, \mathbf{z}|\mathbf{x}, \theta^y, \psi) = \frac{1}{U^y(\mathbf{x})}\prod_{t=1}^{T}\exp\Big(\sum_k\Big(\underbrace{\theta_k^y f_k(y_{t-1}, y_t, x_t)}_{\text{task(y) factor}} +$$
$$\underbrace{\psi_k f_k(y_t, z_t, x_t)}_{\substack{\text{label dependency}\\ \text{factor}}}\Big)\Big). \quad (5)$$

Here

$$U^y(\mathbf{x}) = \sum_{\mathbf{y},\mathbf{z}}\prod_{t=1}^{T}\exp(\sum_k(\theta_k^y f_k(y_{t-1}, y_t, x_t) + \psi_k f_k(y_t, z_t, x_t))).$$

We can write a similar model for the task z. In this model, first type of clique depends on the adjacent labels from task z along with x_t i.e. (z_t, z_{t-1}, x_t) while the other type of clique is similar to the model for task y.

$$p^z(\mathbf{y}, \mathbf{z}|\mathbf{x}, \theta^z, \psi) = \frac{1}{U^z(\mathbf{x})}\prod_{t=1}^{T}\exp\Big(\sum_k\Big(\underbrace{\theta_k^z f_k(z_{t-1}, z_t, x_t)}_{\text{task(z) factor}} +$$
$$\underbrace{\psi_k f_k(y_t, z_t, x_t)}_{\substack{\text{label dependency}\\ \text{factor}}}\Big)\Big). \quad (6)$$

Remarks: Two main points to be noted about these two SHARED models are: (a) though we have two models, one for each task, each of these models is sufficient to produce labels for both tasks, and (b) the parameters θ^x, θ^y are task specific while parameters ψ are common to both tasks which facilitates learning transfer among both tasks. Since each model can be used to produce labels for both tasks, these two tasks can be thought as two views, and one can use co-training with these models to build a semisupervised model.

DEFINITION 2. *A* SHARED *multilabel model is a set of task-specific models, where each task-specific model is defined as in* (4) *but all parameters corresponding to the label-dependency factor are same. In other words:*

$$\psi^{y_1} = \psi^{y_2} = \ldots = \psi^{y_k} = \psi.$$

Next we construct our objective function to fit data to these models. We take four specific approaches to define objective function as described below.

Joint Optimization:.
We hypothesize that although each of these models are sufficient to learn the labels for both tasks independently, it will be advantageous to learn them simultaneously. Consequently, we define a joint model that is the product of both models[2]. We maximize the likelihood of the data under this model, i.e., find the parameters by optimizing the joint log likelihood. This is equivalent to minimizing the loss on the training data. To reduce the overfitting, we define Gaussian prior with mean $\mu = 0$ and covariance matrix

[2]Note that though each of these two models gives us a probability distribution over $(\mathbf{y}, \mathbf{z})$, product of these two models is not a probability distribution. This product is taken only to facilitate the joint learning – a practice used in MTL [24, 4, 6]. One can also think of maximizing this joint log likelihood as minimizing the *cumulative* loss of both models on the training data which is the negative of joint log likelihood.

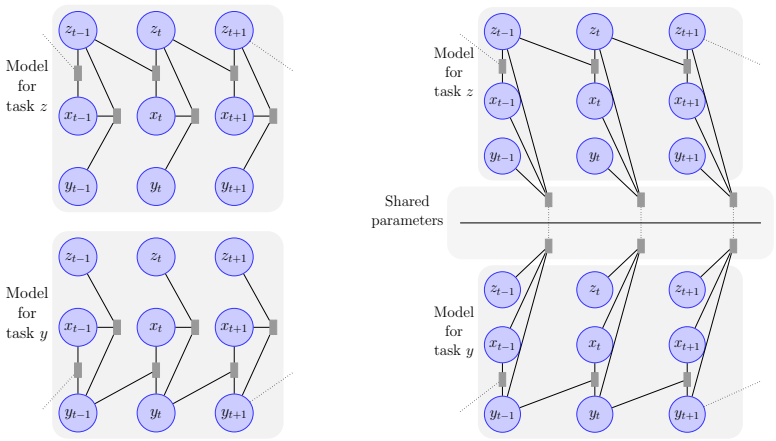

Figure 2: Graphical model representations of UNSHARED (left) and SHARED (right) models. Note the common factors in the SHARED (right) model, above and below the horizontal line. These factors are defined over the same random variables and share the parameters.

$\Sigma = I/\eta$ for all parameters i.e., $p(\theta^y) \propto \exp(-\frac{\eta^y}{2}\|\theta\|^2)$, $p(\theta^z) \propto \exp(-\frac{\eta^z}{2}\|\theta\|^2)$ and $p(\psi^y) \propto \exp(-\frac{\eta^o}{2}\|\theta\|^2)$. The log likelihood of the data with this modeling approach can be written as:

$$\begin{aligned}\ell(\theta^y, \theta^z, \psi) = &\sum_{i=1}^{n} \log p^y \left(\mathbf{y}^{(i)}, \mathbf{z}^{(i)} | \mathbf{x}^{(i)}, \theta^y, \psi\right) \\ &+ \log p^z \left(\mathbf{y}^{(i)}, \mathbf{z}^{(i)} | \mathbf{x}^{(i)}, \theta^z, \psi\right) \\ &- \frac{\eta^y}{2}\|\theta^y\|^2 - \frac{\eta^z}{2}\|\theta^z\|^2 - \frac{\eta^o}{2}\|\psi\|^2 \end{aligned} \tag{7}$$

The derivatives of the above joint log likelihood with respect to θ^y (similar for θ^z) and ψ are:

$$\begin{aligned}\frac{\partial(\ell)}{\partial\theta^y_k} = &\sum_i \sum_t f_k(y^{(i)}_t, y^{(i)}_{t-1}, x^{(i)}_t) \\ &- \sum_i \sum_t \sum_{y,y'} p^y(y, y' | \mathbf{x}^{(i)}, \theta^y, \psi) f_k(y, y', x^{(i)}_t) - \eta^y \theta^y_k \end{aligned} \tag{8}$$

$$\begin{aligned}\frac{\partial(\ell)}{\partial\psi_k} = &\sum_i \sum_t 2 f_k(y^{(i)}_t, z^{(i)}_t, x^{(i)}_t) \\ &- \sum_i \sum_t \sum_{y,z} p^y(y, z | \mathbf{x}^{(i)}, \theta^y, \psi) f_k(y, z, x^{(i)}_t) \\ &- \sum_i \sum_t \sum_{y,z} p^z(y, z | \mathbf{x}^{(i)}, \theta^z, \psi) f_k(y, z, x^{(i)}_t) - \eta^o \psi \end{aligned} \tag{9}$$

where, the first term is simply the feature value while the second (and third) terms are the expectations of the feature values over all possible label combinations, as is standard in log-linear models [9, 20]. Observe that computing these expectations require us to compute marginal probabilities, i.e., $p^y(y, y' | \mathbf{x}^{(i)}, \theta^y, \psi)$, $p^y(y, z | \mathbf{x}^{(i)}, \theta^y, \psi)$ and $p^z(y, z | \mathbf{x}^{(i)}, \theta^z, \psi)$.

Note that the joint likelihood function $\ell(\theta^y, \theta^z, \psi)$ is convex in all its parameters i.e. θ^y, θ^z and ψ and hence can be optimized by a number of techniques. In our implementation, we use L-BFGS which has previous shown to outperform other techniques [28]. For inferences, we need two kind of inferences, one for computing marginals, e.g., $p^y(y, y' | \mathbf{x}^{(i)}, \theta^y, \psi)$ (sum-inference) and other for computing the most likely label i.e., $\arg\max_{\mathbf{y},\mathbf{z}} p(\mathbf{y}, \mathbf{z} | \mathbf{x})$ (max-inference). We use belief propagation for sun-inferences and Viterbi for max-inferences.

The above described model has some resemblance with the factorial CRF model[31] (described in Section 4) with the important difference that the factorial CRF has one single model which is jointly optimized for all tasks and, therefore, has no explicit parameter sharing. On the other hand, we break the factorial CRF in two separate tasks and then explicitly share the parameters among both tasks. This difference is important because breaking the one model into two models increases their discriminative power (the normalization factor is also broken). Such a separate framework allows the transfer of learning through parameter sharing but at the same time, leaves enough room for independent learning. This independent learning is important as you shall see in the experiments, in some cases, UNSHARED model performs better than the factorial CRF because in those cases independent learning is more important that the partial sharing as done in factorial CRF. For mathematical details on this, refer to Appendix A.

3.2 Variance Models

The primary purpose of multitask learning framework is to be able to transfer learning among multiple tasks in a way that each model is able to model its own task, and at the same time, is also able to benefit from other tasks. We have thus far, incorporated this paradigm by having a set of parameters common among different tasks. The task specific part of each task is captured by a factor specific to that task. We extend this framework by splitting the common set of parameters (label dependency) into two parts: one task specific while other common. We hypothesize that the whole label dependency factor may not be common to both tasks, but only a part of it. As we shall see shortly that it will bring flexibility in the model, allowing one to control the amount of transfer among different tasks.

Along the lines of [24], we believe that the parameters corresponding to the label dependency factor lie around a common set of parameters having their own variance specific to task. With this assumption, the common set of parame-

ters ψ can be written as:

$$\psi^y = \psi^o + \nu^y$$

Now, ψ^o is the part that is common to all tasks while ν^y is the task specific part. This is to indicate that there might be a component of ψ that is only specific to that task when considering parameters ψ. The task y model under this assumption can be written as following (task z model will be similar):

$$p^y(\mathbf{y}, \mathbf{z}|\mathbf{x}, \theta^y, \nu^y, \psi^o) = \frac{1}{U^y(\mathbf{x})} \prod_{t=1}^{T} \exp\Big(\sum_k \Big(\theta_k^y f_k(y_{t-1}, y_t, x_t) + \nu_k^y f_k(y_t, z_t, x_t) + \psi_k^o f_k(y_t, z_t, x_t) \Big) \Big)$$

The log likelihood under this model can be given as (with Gaussian prior on each set of parameters):

$$\ell_y(\theta^y, \nu^y, \psi^o) = \sum_{i=1}^{n} \log p^y(\mathbf{y}^{(i)}, \mathbf{z}^{(i)}|\mathbf{x}^{(i)}, \theta^y, \nu^y, \psi^o) - \frac{\eta^y}{2}\|\theta^y\|^2 - \frac{\lambda}{2}\|\nu^y\|^2 - \frac{\eta^o}{2}\|\psi^o\|^2$$

We emphasize here the importance of the weight factor λ associated with the regularization $\|\nu\|^2$. This λ enforces the model to share the label dependency parameters among different tasks. A high value of λ/η^o means that there will be more sharing among tasks while a small value of λ/η^o means that the task would be unrelated as if there is no sharing of parameters among tasks. Note that when $\lambda/\eta^o \to 0$, it will force parameters ψ^o to go to zero which will results in an UNSHARED model; and when $\lambda/\eta^o \to \infty$, it will force task-specific parameters θ^y and ν^y to go to zero which will result in a model that will be completely shared, i.e., same for all tasks. Therefore, one can also think of this λ factor as an interpolating factor, interpolating between completely shared model and an UNSHARED model.

4. RELATED WORK

The work done in this paper relates to two streams of literature, one in dialogue-act/issue-status classification, and other in multitask learning. To the best of our knowledge, there has been no attempt to apply multitask learning framework to customer-care domain for predicting dialogue acts and issue status. Most of the work in the dialogue act classification has been done in spoken dialogue domain, and has mainly focused on either feature engineering aspect of the problem or experimenting with various classification techniques. In spoken dialogue systems, Samuel et al. [27] and Jurafsky et al. [18] focus on lexical and syntactic features, Julia et al. [17] and Rangarajan Sridhar et al. [25] focus on acoustic and prosodic features, while Louwerse and Crossley [22] and Bangalore et al. [8] focus on using various n-gram features. In the written dialogue systems, Forsyth [11] used keyword based approaches to classify dialogue acts, and Ivanovic [14] used n-gram based features. In the class of experimenting with various classification techniques, various methods such as Hidden Markov Model [29], Naive Bayes [14, 11], maximum entropy model [15], support vector machine [15] have been applied.

In multitask leaning, most of the work has focused on the standard classification or regression problems, a very few have focused on the sequence labeling problem, e.g. [30]. In this work, authors do not use MTL setting directly. They learn each task independently but in a cascaded manner i.e. use the output of one task as input to the other, but tests by considering all tasks simultaneously. Therefore, authors do not make use of tasks' relatedness or label dependency at the training time. In classification and regression MTL, there are mainly two approaches, Bayesian and non-Bayesian. In both the approaches, one of the fundamental problem is defining the task relatedness and then incorporating that in the model. In MTL literature, most of the existing methods first assume a structure that defines the task relatedness, and then incorporate this structure in the MTL framework in the form of a regularizer [6, 24, 16]. There are many other approaches to multitask learning such as subspace methods [4, 6, 5], parameter proximity methods [24], and task clustering methods [16]. In subspace method, it is assumed that the parameters of different tasks lie in a subspace whereas in proximity method, we assume that task parameters w_t for each task is close to some common task w_0 with some variance v_t. These v_t and w_0 are learned by minimizing the *Euclidean* norm which is again equivalent to working in the linear space. This idea of proximity method is later generalized through manifold regularization [1] and clustering [16].

For the sake of completeness we give a brief description of Factorial CRF, which will also be our primary baseline. Factorial CRF [31] model is an extension of linear-chain CRFs that repeat structures and parameter over sequences. If we denote by $\Phi_c(y_{c,t}, x_t)$ the repetition of clique c at time step t, then a factorial CRF defines the probability of a label sequence $\mathbf{y}$ given the input $\mathbf{x}$ as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{\prod_t \Phi_c(\mathbf{y}_{c,t}, \mathbf{x}_t)}{\mathbf{Z}(\mathbf{x})}$$

Factorial CRF can be generalized to model connection between multiple label sequences, i.e. $\mathbf{y}_l$ for $l = \{0, 1, .., L\}$ for the same input sequence $\mathbf{x}$. Sutton, et al., [31] defines the $p(\mathbf{y}|\mathbf{x})$ distribution as below:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{\mathbf{Z}(\mathbf{x})} \left(\prod_{t=1}^{T-1} \prod_{l=1}^{L} \Phi_l(y_{l,t}, y_{l,t+1}, \mathbf{x}, \mathbf{t}) \right) \left(\prod_{t=1}^{T} \prod_{l=1}^{L-1} \Psi_l(y_{l,t}, y_{l+1,t}, \mathbf{x}, \mathbf{t}) \right)$$

where $\Phi_l()$ is the task specific factor that models the dependency between the consecutive tokens in a label sequence whereas $\Psi_l()$ is the label dependency factor that models the dependency among labels of the same token. Although factorial CRFs model the dependencies among multiple labels in a sequence, it considers the whole learning problem as one single task. Enforcing this one task structure on the problem constrains the problem, leaving little room for independent learning from multiple labeling tasks. On the other hand, in our method, we break the problem into multiple tasks, allowing room for flexibility for independent learning from both label dependency factor Ψ_l and task specific factor Φ_l, and at the same time benefiting from each other through explicit parameter sharing.

Another line of work [13, 12] in which authors attempt to use MTL framework to sequential tasks, is not applicable to our setting because this framework does not really consider multiple tasks as given in our problem, rather, it artificially creates multiple tasks by considering auxiliary tasks following the work of Ando and Zhang [2]. Dhillon et al. [10] proposes another model for structured prediction

tasks (web structure) which falls into weight regularization class of multitask learning methods. Unlike our method, this method does not exploit the correlation between two labels, and neither does it take advantage of the fact that both label sequences belong to the same example sequence. Furthermore, this method is particularly designed for web-information extraction since it uses the web graph structure for regularizing multiple tasks, and therefore, is not applicable to our setting.

5. EXPERIMENTS

In this section, we describe the datasets, our experimental methodology, and report results.

5.1 Dataset

We evaluate and report our results on two datasets. The first dataset comes from an electronic conversation medium over social media (twitter). The example set is borrowed from real conversations (chat) between customers and customer care agents for a particular telecommunication carrier. Two specific tasks are designed in this case where the chat sentences are labeled for (1) nature of dialogue between customer and agent (namely *Dialogue Act*), and (2) nature of the state of the issue being discussed by customer and agent (namely *Issue Status*). We employed 3 annotators for labeling each sentence present in the conversations. Each conversation is treated as a sequence example akin to a sentence in the first dataset. For first task, sentences are annotated from 12 label as given in Table 3. For second task, sentences are annotated with 4 labels: Open Issue, Issue Resolved, Change Medium of Communication, and Issue Closed, as shown in Table 2. We take 291 annotated conversations with a total of 3072 sentences with 10.6 sentences per conversation. We append frequent bigrams, emoticons, punctuation and standard word features such as capitalization etc.

In order to show the effectiveness of our method beyond issue-status and dialogue act prediction problems, we also experiment with a second dataset. This second dataset corresponds to a noun phrase chunking and POS tagging tasks, and comes from a CoNLL 2000 shared-task [3]. We take a smaller set of the original data set primarily because MTL only makes sense when single task learning (STL) is not sufficient (i.e. it is difficult). This difficulty of STL can be attributed to two main reasons– one, there are not enough labeled examples, and second, the problem itself is a difficult problem despite being enough labeled examples. The CoNLL dataset violates both of these conditions, i.e., there are enough labeled examples, and these labeled examples give a very good accuracy i.e., in the range of 99%. So in order to make the MTL applicable here, we increase the difficulty of the problem by reducing the size of labeled data. The smaller dataset consists of total 350 sentences containing 8785 individual tokens as examples. We split the data into 150 train and 200 test examples. In this dataset, two tasks correspond to the NP chunking and part-of-speech (POS) tagging. The idea is to get performance improvement by learning from these two tasks simultaneously. This dataset is also used in the baseline method by Sutton et al., [31]. For the sake of completeness, we also ran our experiments on full dataset, and all methods performed between 98% and 99%.

[3]Publicly available at [32] `http://mallet.cs.umass.edu/grmm/data`

5.2 Models Comparisons

We use following models for comparisons. Among these models, one is baseline, other models are ours, with different variations.

- **Factorial CRF[31]:** We use this as our primary baseline.
- **Unshared model:** Both tasks have their own separate parameters (See Definition 1).
- **JOSP:** (Jointly Optimized Shared Parameters) This is the shared model where parameters are learned by optimizing the joint likelihood.
- **AOSP:** (Alternatively Optimized Shared Parameters) This is the shared model but in contrast to the joint optimization, here parameter are learned in an alternative fashion, i.e., we split the joint likelihood into two parts, one for each task and optimize the parameters alternatively. ψ is still a common set of parameters among both tasks however we do not optimize the joint likelihood.
- **JOVM:** (Jointly Optimized Variance Model) Variance model as defined in Section 3.2 but parameters are learned by optimizing the joint likelihood.
- **AOVM:** (Alternatively Optimized Variance Model) Variance model as defined in Section 3.2 but parameters are learned alternatively.

5.3 Results

We use accuracy as our metric of evaluation. Here we define accuracy as fraction of correctly labeled tokens in sequences present in the test set. It is important to note that we report the accuracy from their respective models i.e., each model gives labels for all tasks but we take the labels from the model that is specific to that task (as described in Section 3.1). The results for the two datasets are presented in Table 4. We vary the training size and report the results. All reported results are averaged over 10 random runs, and their means and standard deviations are reported. For the baseline, we use the code provided by the authors. All the hyper-parameters are tuned via cross validation with 10 folds.

From these results we draw multiple conclusions: (1) In general, learning tasks together in MTL setting —either directly or using variance method— helps. All results show significant improvement over factorial CRF. This improvement is higher when there are fewer labeled examples. (2) Though in some cases, MTL (Shared model and Variance model) helps over factorial CRF but learning them independently (UNSHARED model) helps even more. e.g. Issue Status task. This establishes the fact that not all tasks improve from MTL. In fact, it shows that in multiple tasks, one task can benefit from other tasks while another cannot.

From the accuracy figures, it can be inferred that the Task 1 is harder than Task 2 for both datasets. The results reported show that the accuracy improvements are greater for Task 1 compared to Task 2. For difficult tasks, results show that learning both tasks independently (UNSHARED model) hurts. Learning them together through explicit parameter sharing gives significant improvement over UNSHARED

Category	Description	Example
Open	When a conversation is ongoing and a further message is expected to follow	@userxyz Hi, that's not good.
Solved	When the current message solves the problem mentioned in the current conversation, without requiring any further message	No, your payment would just increase by $5 a month and you will keep your Shrinkage milestone.
Closed	When the current message closes the conversation, even if the problem is not solved	@user123 We would rally hate to see you go.
Change Channel	When the CRM agent asks the customer to change channel by sending an email or a direct message. In this case, the conversation halts since further exchanges are on other private channels.	Can you please email me directly at xyz@abc.com and I will gladly look into this.

Table 2: Categories to label the engagement status of tweets

Category	Description	Example
Complaint	When a customer complains	@vmucare IâĂŹve sent an email, but I am absolutely disgusted with the customer care I am receiving
Apology	When an agent apologies	@kristenmchugh22 I do apologize for the inconvenience.
Answer	When someone answers a request	@BoostCare yea, allow my texts and calls to go out
Receipt	When someone acknowledges receipt	@VMUcare ok
Compliment	When someone sends a compliment	I still love VM and my intercept
Response to positivity	When someone acknowledges a previous positive message	@harryruiz No problem!
Request	When someone requests information	Please help me out.
Greeting	Greetings	@LucusHughes13 Hi there!
Thank	When someone expresses thanks	Thank you for being so patient.
Announcement	When a customer announces an information	@VMUcare phone stolen last night
Solved	When a message clearly states that a problem is solved	Close one!
Other	Any other message	Wow!

Table 3: Categories to label tweets in conversations according to the linguistic theory of Conversation Analysis

or factorial CRF. This observation along with the observation that MTL improvement is higher when there are fewer labeled examples, provide evidence in support of the hypothesis about the applicability of MTL, i.e., MTL is applicable when the underlying problem is difficult, either inherently or because of the scarcity of labeled examples. The results are not as clear for Task 2, but still, in these tasks, results indicate that one should use MTL – either learn all tasks together through *explicit* parameter sharing (Shared model or Variance model) or not share anything at all (UNSHARED MODEL). Partial sharing (one task structure) as in factorial CRF gives inferior results.

We also varied the training size and recorded the accuracies. The results are plotted in Figure 3 and Figure 4, for CoNLL and conversation data respectively. These figure also support our earlier hypothesis. From these figures, we see that when tasks are difficult (Task 1), MTL models (variance and shared) perform better, but when tasks are rather easy, UNSHARED model performs better.

Convergence of algorithms: In Figure 5, we plot the negative likelihood of the data as algorithm progresses in order to show the convergence of different variations of the algorithm. These figures reveal that all variations of the algorithm converge. Note that although the alternate optimization method is not theoretically guaranteed to converge, it converges in practice, as seen in experiments across the board. Because of convexity, in joint optimization, one should expect a monotonic decrease in the likelihood, however, we notice that this is not the case, and there are some irregularities in the figures. These irregularities are the artifact of the *optimization* algorithm (i.e. BFGS) – sometimes the optimization step does not move in the right gradient direction and therefore needs restarting.

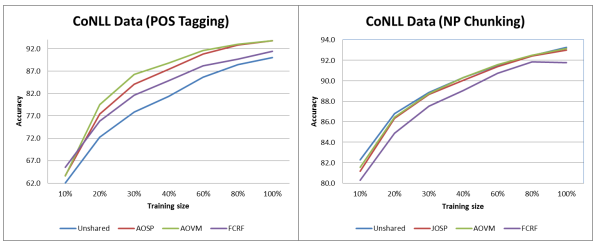

Figure 3: Variation with training size for CoNLL data

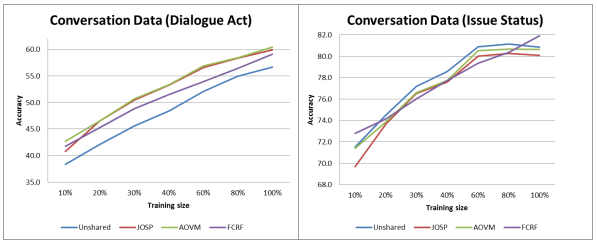

Figure 4: Variation with training size for conversation data

6. CONVERSATION ANALYSIS SYSTEM

To appreciate the effectiveness of dialogue act and issue status predictions in a broader system-level user engagement context, we present a social-CRM system use case. A traditional CRM system handles customer care requests in a reactive manner where customer describes her issue via any communication medium and CRM system offers potential solutions. In contrast, a social-CRM system takes a proactive approach where a listening component filters various social media streams to spot users mentioning issues with the respective product/service. The social-CRM agent then engages the user into conversation regarding her issue and offers potential solutions via social media itself. In such a proactive system, engagements are of asynchronous nature and often takes place among many agents to many users, which makes it hard for keeping track of the progress of issue resolutions. Therefore, an automated solution for effectively tracking conversations is of paramount importance.

Dialogue act and issue status of conversations can provide an effective solution for automating the issue tracking and various other qualitative and quantitative metrics that a customer care center is typically interested in. For illustration, we present the outline of an automated social-CRM

Table 4: Experimental results for MTL for CoNLL and Social Conversations datasets

Dataset	Task	%Train	MTL					DCRF
			JOVM	AOVM	JOSP	AOSP	Unshared	
CoNLL	POS Tagging (Task 1)	(30%)	86.0 ± 1.3	**86.3 ± 1.2**	83.7 ± 1.6	84.1 ± 1.4	77.9 ± 1.2	81.6 ± 1.4
		(60%)	91.5 ± 0.5	**91.6 ± 0.4**	90.7 ± 0.5	90.8 ± 0.6	85.7 ± 0.4	88.2 ± 0.5
	NP Chunking (Task 2)	(30%)	**89.0 ± 0.4**	88.8 ± 0.9	88.5 ± 1.1	88.7 ± 0.9	88.8 ± 0.9	87.5 ± 0.8
		(60%)	91.5 ± 0.5	**91.6 ± 0.3**	91.3 ± 0.5	91.4 ± 0.3	91.5 ± 0.4	90.7 ± 0.4
Social Conversation	Dialogue Act (Task 1)	(30%)	**51.4 ± 2**	50.7 ± 1.4	45.3 ± 2	50.5 ± 2	45.6 ± 2.0	48.9 ± 1.1
		(60%)	56.7 ± 2.6	**56.9 ± 1.8**	55.7 ± 2.8	56.6 ± 1.6	52.1 ± 1.9	53.9 ± 1.2
	Issue Status (Task 2)	(30%)	**77.2 ± 0.9**	76.6 ± 0.8	74.4 ± 2.9	76.5 ± 1.1	**77.2 ± 1.1**	76.0 ± 1.4
		(60%)	80.3 ± 1.1	80.5 ± 1.2	80.8 ± 1.5	80.0 ± 1.1	**80.9 ± 0.6**	79.4 ± 0.5

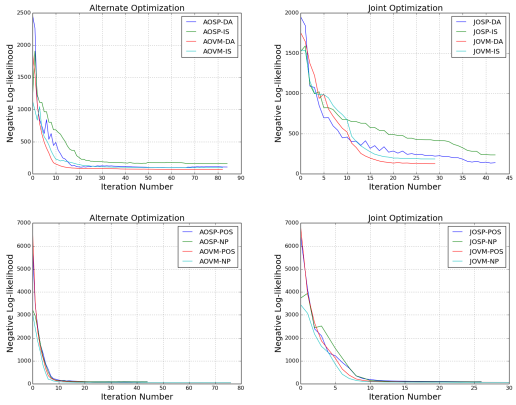

Figure 5: Convergence of different algorithms on Social Conversation (top row) and CoNLL (bottom row) datasets.

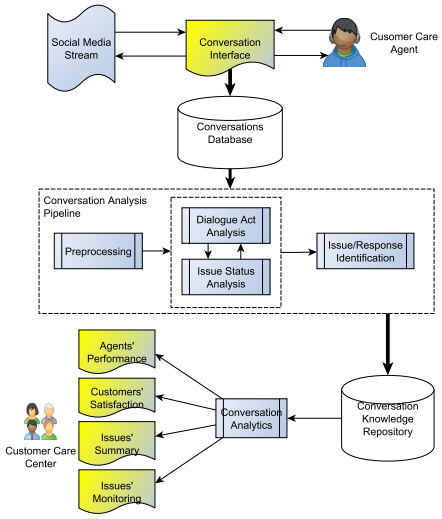

Figure 6: System Architecture

system in Figure 6 where conversation analysis comprising of dialogue act and issue status prediction is an integral part of the system. The proposed automated system can be broken into further functional units where directional arrows represent data flow among components.

Conversation Front-end corresponds to the listening phase where a keyword (or semantics) based filtering spots users having issues with the product/service. All the conversations resulting from customer agent interaction become input to the conversation analysis phase.

Conversation Analysis Pipeline corresponds to dialogue act and issue status identification phase. After generating dialogue act and issue status labels (details in Section 3), conversations that are about any issues are filtered for next phase. All the conversations containing either of `COMPLAINT, REQUEST, ANNOUN` or `OPEN` belong to this set.

Conversation Analytics corresponds to "reporting and further action" phase where various typical customer care performance metrics are employed to (1) measure the effectiveness of social-CRM; and (2) identify issues that need further follow up. Based upon dialogue act and issue status labels, we can derive various effectiveness measures that can be easily consumed in already existing performance metrics of any CRM system. Following are a few categories of such metrics:

(a) Issue Monitoring: Several metrics that characterize the current state of a conversation fall in this category. (i) *Issue resolution rate* can be defined as fraction of the conversations whose first message is labeled as `OPEN` and last as `SOLVED`. In our dataset (Section 5), there were a total of 56.61% such cases. (ii) *Properly handled conversations* can be defined as fraction of the conversations whose first message is labeled as `OPEN` and last as either of `SOLVED, CLOSED, CHANGE CHANNEL`. In our dataset, we found 77.62% such cases. (iii) *Assistance conversations* can be defined as fraction of the conversations whose first message was labeled `OPEN` and `REQUEST`. In our dataset, we found 31.18% such cases.

(b) Issues Summary: The conversation sentences that contain either of `COMPLAINT, REQUEST, ANNOUN` and `OPEN`, can be assumed primary issues faced by social media users. Further clustering of these tweet sentences[26] can identify types of issues processed by social-CRM system.

(c) Customer Satisfaction: (i) *Customer conversion rate* can be defined as fraction of the conversations whose first message was labeled as `COMPLAINT or REQUEST` and last as either of `THANKS, RESPOS, SOLVED, COMPLIMENT, ANSWER`. Total cases: 36.61%(ii) *Customer "hang-up" rate* can be defined as fraction of the conversations whose tail messages are labeled as `COMPLAINT or REQUEST` and no message as `THANKS, RESPOS, COMPLIMENT`. Total cases: 34.57%

(d) Agent's Performance: agent's performance metrics can be derived similarly by combining issues' monitoring and customers' satisfaction rate on per agent basis.

7. CONCLUSION AND FUTURE WORK

In this paper, we have presented a novel method for learning from multiple sequence labeling tasks, in particular for issue status and dialogue act prediction tasks for social media user engagement. Unlike the previous methods, our method models each task as one single model, but still transfer the learning from other tasks through parameters sharing. We have shown through various experiments on two datasets that our method consistently outperforms the state-

of-the-art method for such tasks, especially in cases when tasks are relatively harder and there are fewer labeled examples. One additional advantage of our method is that unlike most methods in MTL in which each model only learns on its own labels (and hence outputs its own labels only), the proposed method learns using all labels which makes this approach extensible for semi-supervised setting through co-training. Since getting labeled data for such supervised classification method is very expensive, we would like to explore semi-supervised techniques to reduce the method's reliance on labeled data. Although our method can naturally be extended for semi-supervised setting since it gives two classifier models, each classifier predicting the labels for both tasks, a experimental study verifying the effectiveness of such method is yet to be done. A further theoretical analysis of understand the framework also remains to be done in future.

8. REFERENCES

[1] A. Agarwal, H. D. Iii, and S. Gerber. Learning multiple tasks using manifold regularization. In *Advances in neural information processing systems*, pages 46–54, 2010.

[2] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6:1817–1853, 2005.

[3] J. Ang, Y. Liu, and E. Shriberg. Automatic dialog act segmentation and classification in multiparty meetings. In *Proc. ICASSP*, volume 1, pages 1061–1064, 2005.

[4] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *NIPS '06*, 2006.

[5] A. Argyriou, T. Evgeniou, M. Pontil, A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. In *Machine Learning*. press, 2007.

[6] A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. In *NIPS '08*. 2008.

[7] C. H. Baird and G. Parasnis. From social media to social customer relationship management. *Strategy & Leadership*, 39(5):30–37, 2011.

[8] S. Bangalore, G. Di Fabbrizio, and A. Stent. Learning the structure of task-driven human–human dialogs. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(7):1249–1259, 2008.

[9] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

[10] P. S. Dhillon, S. Sellamanickam, and S. K. Selvaraj. Semi-supervised multi-task learning of structured prediction models for web information extraction. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 957–966. ACM, 2011.

[11] E. N. Forsyth. *Improving automated lexical and discourse analysis of online chat dialog*. PhD thesis, Monterey, California. Naval Postgraduate School, 2007.

[12] S. He, X. Wang, Y. Dong, T. Zhang, and X. Bai. Multi-task learning in conditional random fields for chunking in shallow semantic parsing. In *PACLIC*, pages 180–189, 2009.

[13] S. He, T. Zhang, X. Bai, X. Wang, and Y. Dong. Incorporating multi-task learning in conditional random fields for chunking in semantic role labeling. In *Natural Language Processing and Knowledge Engineering, 2009. NLP-KE 2009. International Conference on*, pages 1–5. IEEE, 2009.

[14] E. Ivanovic. Dialogue act tagging for instant messaging chat sessions. In *Proceedings of the ACL Student Research Workshop*, pages 79–84. Association for Computational Linguistics, 2005.

[15] E. Ivanovic. *Automatic instant messaging dialogue using statistical models and dialogue acts*. University of Melbourne, Department of Computer Science and Software Engineering, 2008.

[16] L. Jacob, F. Bach, and J.-P. Vert. Clustered multi-task learning: A convex formulation. In *NIPS '08*, 2008.

[17] F. N. Julia, K. M. Iftekharuddin, and A. U. ISLAM. Dialog act classification using acoustic and discourse information of maptask data. *International Journal of Computational Intelligence and Applications*, 9(04):289–311, 2010.

[18] D. Jurafsky, E. Shriberg, B. Fox, and T. Curl. Lexical, prosodic, and syntactic cues for dialog acts. In *Proceedings of ACL/COLING-98 Workshop on Discourse Relations and Discourse Markers*, pages 114–120, 1998.

[19] S. N. Kim, L. Cavedon, and T. Baldwin. Classifying dialogue acts in one-on-one live chats. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 862–871. Association for Computational Linguistics, 2010.

[20] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[21] Q. Liu, X. Liao, H. L. Carin, J. R. Stack, and L. Carin. Semisupervised multitask learning. *IEEE 2009*, 2009.

[22] M. M. Louwerse and S. A. Crossley. Dialog act classification using n-gram algorithms. In *FLAIRS Conference*, pages 758–763, 2006.

[23] D. Marcheggiani, O. Täckström, A. Esuli, and F. Sebastiani. Hierarchical multi-label conditional random fields for aspect-oriented opinion mining. In *Advances in Information Retrieval*, pages 273–285. Springer, 2014.

[24] C. A. Micchelli and M. Pontil. Regularized multi-task learning. In *KDD 2004*, pages 109–117, 2004.

[25] V. K. Rangarajan Sridhar, S. Bangalore, and S. Narayanan. Combining lexical, syntactic and prosodic cues for improved online dialog act tagging. *Computer Speech & Language*, 23(4):407–422, 2009.

[26] A. Rangrej, S. Kulkarni, and A. V. Tendulkar. Comparative study of clustering techniques for short text documents. In *Proceedings of the 20th international conference companion on World wide web*, pages 111–112. ACM, 2011.

[27] K. Samuel, S. Carberry, and K. Vijay-Shanker. Dialogue act tagging with transformation-based learning. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 1150–1156. Association for Computational Linguistics, 1998.

[28] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 134–141. Association for Computational Linguistics, 2003.

[29] A. Stolcke, N. Coccaro, R. Bates, P. Taylor, C. Van Ess-Dykema, K. Ries, E. Shriberg, D. Jurafsky, R. Martin, and M. Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Comput. Linguist.*

[30] C. Sutton and A. McCallum. Composition of conditional random fields for transfer learning. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 748–754. Association for Computational Linguistics, 2005.

[31] C. Sutton, A. McCallum, and K. Rohanimanesh. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *The Journal of Machine Learning Research*, 8:693–723, 2007.

[32] E. F. Tjong Kim Sang and S. Buchholz. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*, pages 127–132. Association for Computational Linguistics, 2000.

[33] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *J. Mach. Learn. Res.*, 8:35–63, 2007.

[34] K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *ICML '05*, 2005.

APPENDIX

A. FACTORIAL CRF AND SHARED JOINT MODEL

For reference, we write below the factorial CRF model:

$$p(\mathbf{y},\mathbf{z}|\mathbf{x},\theta^y,\theta^z,\psi) = \frac{1}{U(\mathbf{x})}\prod_{t=1}^{T}\exp\Bigg(\sum_k\Big(\underbrace{\theta_k^y f_k(x_t,y_{t-1},y_t)}_{\text{task(y) factor}} + \underbrace{\theta_k^z f_k(x_t,z_{t-1},z_t)}_{\text{task(z) factor}} + \underbrace{\psi f_k(x_t,y_t,z_t)}_{\text{label dependency factor}}\Big)\Bigg)$$

It might look like that the above factorial CRF model is similar to the product of the two models p_y and p_z which it is not. This is because of the normalization factor in individual model. The product of two models ((5) and (6))can be written as:

$$\begin{aligned} q(\mathbf{y},\mathbf{z}|\mathbf{x},\theta^y,\theta^z\psi) &= p^y(\mathbf{y},\mathbf{z}|\mathbf{x},\theta^y,\psi)\; p^z(\mathbf{y},\mathbf{z}|\mathbf{x},\theta^z,\psi) \\ &= \frac{1}{U'(\mathbf{x})}\prod_{t=1}^{T}\exp\Bigg(\sum_k \theta_k^y f_k(x_t,y_{t-1},y_t) \\ &\qquad + \theta_k^z f_k(x_t,z_{t-1},z_t) + 2\psi_k f_k(x_t,y_t,z_t)\Bigg) \end{aligned}$$

where

$$\begin{aligned} U'(\mathbf{x}) &= U^y(\mathbf{x})\; U^z(\mathbf{x}) \\ &= \Bigg(\sum_{\mathbf{y},\mathbf{z}}\prod_{t=1}^{T}\exp\Bigg(\sum_k\Big(\theta_k^y f_k(x_t,y_{t-1},y_t)+\psi_k f_k(x_t,y_t,z_t)\Big)\Bigg)\Bigg) \\ &\qquad \Bigg(\sum_{\mathbf{y},\mathbf{z}}\prod_{t=1}^{T}\exp\Bigg(\sum_k\Big(\theta_k^z f_k(x_t,z_{t-1},z_t)+\psi_k f_k(x_t,y_t,z_t)\Big)\Bigg)\Bigg) \\ &\neq U(\mathbf{x}). \end{aligned}$$

Note that in the above product of the two models, the numerator is very similar to the factorial CRF, (they are the same except that the common factor ψ is counted twice) but the denominator is completely different. The denominator in factorial CRF *cannot* be written as the product of the denominator of two models, i.e., $U'(\mathbf{x}) \neq U(\mathbf{x})$. This breaking of numerator is important because it allows the model to break into multiple tasks hence allowing for independent learning, at the same time facilitating transfer learning through parameter sharing.