A Graph Convolutional Approach to Answer Selection in Community Question Answering

Anonymous Author(s)

ABSTRACT

This paper addresses the question of identifying appropriate user-generated content in social networks. The question is challenging since the "best" content is determined in relation to other content; furthermore, the social interaction graph also influences the identification of content. The general problem is important to Community Question Answer (CQA) platforms, learning to rank scenarios and recommender systems. In contrast to feature-driven methods, we develop a novel data induced relational graph convolutional framework to address this question. We make three contributions. First, we introduce a novel idea of using abstract relationships—contrast, similarity, reflexive—to induce different relational graphs or views on content generated in a social network. The relational graphs can vary from the underlying social interaction graph. Second, we show how to operationalize each relational view with a Graph Convolutional Network (GCN) architecture. Finally, we show through extensive empirical results that boosting techniques improve learning in our convolutional model. This is a surprising result since much of the work on neural architectures develops stacking, fusion or aggregator architectures to combine data modalities. We conduct extensive experiments with excellent experimental results on two different CQA forums—StackExchange and Reddit. Our model improves on an average by over 4% and 16% accuracy in identifying quality content over state-of-the-art baselines for StackExchange and Reddit datasets respectively.

KEYWORDS

Community Question Answering, Graph Convolution, Multi-View Learning, Induced Relational Views

1 INTRODUCTION

A fundamental challenge in user generated content in social networks is: how to identify appropriate content, in response to a query (a search problem); or in response to user behavior (a recommendation problem). The problem is hard for two reasons: first, the "optimal" content is determined *in relationship* to other pieces of content in the network; second, the interaction structure amongst the participants of the social network influences identification of appropriate content. The problem is especially important to Community Question Answer platforms such as StackExchange, where individuals visit to seek answers to nuanced questions, not easily available on prominent search engines.

While there are clear connections to well known Learning to Rank problems in the IR community [2–4], there is only limited work in the context of identification of "best content" among user generated content on social networks using relational information. Induced relational views (or graphs) capture the inter-connected structure of user generated content with potentially varying view semantics, for instance a similarity view would seek to cluster similar content. On the other hand, a contrastive view would seek to connect popular or successful content against other competing submissions. Recent work in this area include work on using Neural techniques on graphs, such as DualGCN [34] to capture multiple modalities of the data. The main limitation of the work in the Deep Convolutional Networks on graphs is the focus on label sharing among network nodes; a concept that is natural to many problem contexts (e.g. computer vision problems), but problematic in the case of identification of appropriate content that also require label contrasts across edges of a graph.

We develop a novel data induced relational framework to address this challenge. The key idea is to use diverse abstract relations—contrastive, similar and reflexive—and operationalize them to *induce* graphs on the content; the induced graph structure depends on the relationship type. The mechanism by which we operationalize the relation may vary; for example, two answers by the same person across two questions may share the same label, if the individual posting the answer has the similar skill contrast with peers who have also posted answers in response to each of the two questions. We design graph convolutional networks (GCNs) appropriate to each relation type and then introduce a boosting framework to combine their outcomes. Our Contributions are as follows:

- **Induced Relational Framework:** We introduced a novel idea of using abstract relationships—contrastive, similar, reflexive—to induce different graphs on content generated in a social network. In contrast, related work typically operate on knowledge graphs with edges of specific relation type (e.g. similarity). The contrastive relation highlights differences between content. We operationalize similarity amongst content created by the same individual; we say that two pieces of content are similar, when same individual who created them is different in a precise sense from other peers who have also contributed competing content. The reflexive relation is the case when a piece of content is judged on its own merit, disregarding competing content. The impact of this framework lies in its ability to induce relation-dependent graphs that can be quite different from the underlying social interaction graph.
- **Operationalizing Relational GCN:** We show how to operationalize each relation to a Graph Convolutional Network architecture. The related work has primarily focused on architectures that support label sharing among network neighbors. We show that the contrastive GCN and the relative-similarity GCN are necessary for addressing our problem.
- **Boosted Architecture:** We show through extensive empirical results that using familiar boosting techniques improves learning in our convolutional model. This is a surprising result since much of the work on neural architectures develops stacking, fusion or aggregator architectures. We show empirically that a boosted 4-layer GCN works better than a 12-layer stacked architecture. In fact, these aggregator

architectures perform worse than the 4-layer Contrastive relational GCN for StackExchange. We conjecture that induced graphs contain noise, since the links amongst nodes are dependent on the data values; stacking results in noise amplification.

We conducted extensive experiments with excellent experimental results on two different CQA datasets—StackExchange and Reddit. The latter is more complicated since unlike StackExchange, it doesn't have answers deemed correct by the questioner. We achieved an improvement of over 4% in accuracy for StackExchange datasets while for Reddit, we achieved a notable gain of around 16% in accuracy over state-of-the-art baselines. We improve by 2% and 7% in MRR for StackExchange and Reddit respectively. Our model is also more robust to label sparsity than other GCN based approaches proposed for multiple relations.

We organize the rest of this paper as follows. In section 2, we formulate our problem statement and then discuss induced relations for Answer Selection problem in section 3. We then detail operationalization of these induced relations in Graph Convolution framework in section 4 and introduce our gradient boosting based aggregator approach in section 5. Section 6 describes experiments. We discuss related work in section 7 and then conclude in section 8.

2 PROBLEM FORMULATION

In Community Question Answer (CQA) forums, an individual who asks a question seeks to identify a good answer from a set of answers to their question. In particular, in StackExchange communities, the questioner can annotate the best answer ("the accepted answer").

> The goal of this paper is to develop a framework to automatically identify the best answer to a question on a CQA forum.

A person picking the best answer could adopt three different strategies. First, she could pick an "accepted answer" in relation to all other answers to the same question. Second, she could may look at the reputation of the person answering the question as a basis, since a person with a high reputation is likely to have earned it by answering other previous questions correctly. Finally, she could ignore all other answers to the question and pick the best answer solely on its own merit.

We view the problem of identifying the best answer a amongst the set of answers $\mathcal{A}_q$ to a question q through the metaphor of a tournament, where players provide answers which compete against each other to win, and where players can compete in multiple tournaments. Each question sets up a single tournament.

Viewed through the lens of the tournament metaphor, there are thus three data induced relationships amongst the answers: contrastive; similar and reflexive. In the contrastive relation, we compare answers in response to the same question, against each other to identify the best one. Thus, the label of an "accepted answer" differs from all other answers to the same question. In the similarity relationship, we examine the similarity amongst answers *by the same player* across different tournaments. Some of these answers will share the same label (e.g. "accepted") if she plays in tournaments where *her skill contrast* with the skill distributions of the competitors are similar. In the reflexive relation, we decide on the tournament winner (i.e. assign a label) without looking at

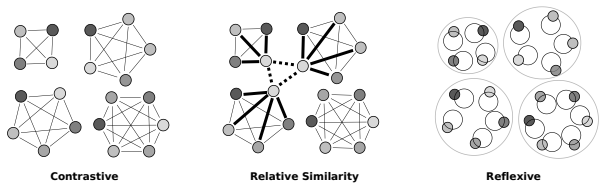


Figure 1: Induced data-driven relational classes; Contrastive, Relative Similarity and Reflexive among (q, a) tuples. Contrastive relation exists between all answers to a question; Relative Similarity connects answers across questions if relative difference in author's skill with other answerers in each question are significant. Reflexive assumes no dependence on other answers for prediction.

the competition. Each induced *view* encodes different relational semantics across the (q, a) tuples. In each view, (q, a) tuples constitute nodes of the induced relational graph, while we add links connecting associated pairs under that *view*.

In the next section, we operationalize the induced *Contrastive*, *Relative Similarity* and *Reflexive* relational graphs of (q, a) tuples for answer selection. We can operationalize a specific relational *view* in more than one way. For instance, relative similarity can be defined on the basis of different node properties. We thus adopt the abstraction of relational classes, where Contrast, Relative Similarity and Reflexive are distinct *relational classes*.

3 INDUCED RELATIONAL VIEWS

In this section, we define and operationalize the three class of *induced relational views* among the (q, a) tuples introduced above. Each (q, a) tuple constitutes a node in each induced relational view of the dataset. We now describe the precise construction of the links in each view. We aim to facilitate feature sharing (with similarity views) and feature discrimination (with contrastive views) among linked nodes.

Let Q be the set of questions and for each $q \in Q$, we associate a set of answers $\mathcal{A}_q$ with it. We associate each answer $a \in \mathcal{A}_q$ with feature $\mathcal{X}_a$, and each question q with feature $\mathcal{X}_q$. Similarly, we associate features $\mathcal{X}_{u,q}, \mathcal{X}_{u,a}$ with the users who author the question q and the answer a. We associate each (q, a) tuple with the aggregated feature $[\mathcal{X}_q, \mathcal{X}_a, \mathcal{X}_{u,q}, \mathcal{X}_{u,a}]$ and a label $y \in \{-1, +1\}$ denoting the answer status (e.g. "accepted answer").

The primary objective of the *Contrastive* class of relational views is to enable the model to discriminate features of accepted answers against competitors. We propose a specific operationalization of the contrast view drawing upon the tournament view of each question in the forum.

Tournament-Style Contrast: For each question $q \in [1, \ldots, Q]$, and each answer $a \in \mathcal{A}_q$, we construct links $(a_1, a_2) \ \forall \ a_1, a_2 \in \mathcal{A}_q$ resulting in a clique of answers to each question. The size of the clique is determined by the size of the answer set $|\mathcal{A}_q|$. Note that in each clique, neighbors differ in their associated labels as one answers has an accepted label while the rest do not. The resulting induced relational graph is a disconnected collection of cliques, representing each tournament. The connections across tournament competitiors (i.e. accepted and non-accepted answers to a question) enables feature contrast when there are significantly

fewer accepted answers compared to the opposite class, an effect we term *Discriminative Feature Magnification*.

The **Relative Similarity** class of views connects answers across different questions (participants across different tournaments) based on their relative features against other candidates. This enables feature sharing across answers that differ in similar ways to their competitors. Consider a simplified example, user Alice participates in four tournaments with variable clique size (i.e. she authors answers to questions with different numbers of other competing answers). Assume a wide skill-margin between Alice and the authors of the other answers in three tournaments. In these cases, it is likely that her answer emerges as the accepted submission or the converse depending on her relative skill. However, results are uncertain in scenarios where the author skill differences are not as pronounced. The relative similarity of Alice's answers (high skill margins) to competing submissions enables us to make a confident prediction.

Hence, one way to operationalize relative similarity is in terms of the author skills. Specifically, the set of answers authored by a specific user are connected together if the magnitude of the relative difference with other competitors in skill levels is the same. We operationalize relative similarity in two ways:

TrueSkill Similarity connects answers authored by a specific user, where the difference in his skill over peers is greater than margin δ. Specifically, if the user authors answers a, a' to questions q, q', we create a link between a and a' if

$$|S_{u,a} - S_{u,b}| > \delta; \forall b \in \mathcal{A}_{(q)}$$
$$|S_{u,a'} - S_{u,c}| > \delta; \forall c \in \mathcal{A}_{(q')}$$

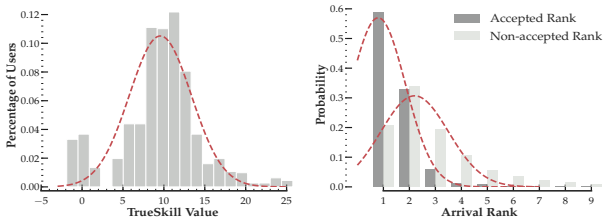
where $S_{u,a}$ is the skill value for the user who authored answer a. Similarly, a link is created for the opposite case when difference is less than $-\delta$. We estimate the user skill values with the TrueSkill rating system [14] computed with their historic performance in the community.

Arrival Similarity The temporal arrival patterns of answers are correlated to their acceptance probabilities (fig. 2). For a specific user authoring answers a, a' to questions q, q', we establish a link between these answers if

$$|T_a - T_b| > \gamma \times \max(T_b); \forall b \in \mathcal{A}_{(q)}$$
$$|T_{a'} - T_c| > \gamma \times \max(T_c); \forall c \in \mathcal{A}_{(q')}$$

where T_a represents the relative time-gap between answer a and the question q. Conversely, we create links when difference is less than $-\gamma \times \max(T_b)$. Our hypotheis is that a similar answering schedule indicates similar user confidence or skill across questions.

The induced *Relative Similarity* views semantically differ from the *Contrastive* view in two ways. First, they enforce label and feature sharing among the connected nodes. In comparison, in the contrastive relationship, the objective is feature discrimination; a node in contrast with its neighbors will have different labels. The differences have important bearing in how we develop convolutional networks. Second, in each tournament, every answer will contrast with its peers. However, since there are multiple ways to operationalize similarity, the corresponding cliques depend on the similarity measure (TrueSkill and Arrival Similarity are two relevant similarity views in CQA). These links are inherently noisy. This is a key factor to consider when we develop our node classification architecture in the next section.



(a) TrueSkill Distribution **(b) ArrivalRank Distribution**

Figure 2: Distribution of the TrueSkill values of users and ArrivalRank of accepted answers and non-accepted answers for the movie StackExchange. Early answers are more likely to be accepted and variance of TrueSkill similarity across users is high.

Finally, **Reflexive** relationships assume prediction of tournament outcomes for each player in isolation as illustrated in fig. 1. In such cases, the label prediction for each (q, a) node depends on the node features alone. This view encompasses most of the feature-driven answer selection literature and methodology [1, 15, 25, 26].

4 INDUCED RELATIONAL GCN

In this section, we introduce Neural Graph Convolution [17] as a node classification approach in our context (section 4.1). We show that the most common formulation adopted in [17] and it's extensions [6, 20] are unsuitable to represent the contrastive class of relational views. We propose a modification to the original formulation to achieve *Discriminative Feature Magnification* between the contrasted nodes (section 4.2). This results in improved separability of classes in the manifold learned by the neural network (further discussion in section 6.7). We then describe the convolution operations for the Relative Similarity views in section 4.3 and Reflexive view in section 4.4.

4.1 Graph Convolution

Graph Convolution models adapt the convolution operations on regular grids (like images) to irregular graph-structured data, learning low-dimensional node representations. The convolution operations on graphs can be described by the product of signal $x \in \mathbb{R}^n$ (feature vectors) with a learned filter g_θ in the fourier domain,

$$g_\theta * x = g_\theta(U\Lambda U^T)x = U g_\theta(\Lambda)U^T x$$

with U denoting the eigenvector matrix of the normalized graph Laplacian $L = I_n - D^{-1/2}AD^{1/2}$ (with the decomposition $L = U\Lambda U^T$). A denotes the adjacency matrix of graph G with n nodes (induced relational graphs in our case with nodes denoting (q, a) tuples), $D_{ii} = \sum_j A_{ij}$ is the corresponding degree matrix.

To circumvent the expensive eigen-decomposition of the Laplacian, Defferrard et al. [5] proposed to approximate $g_\theta(\Lambda)$ with truncated Chebyshev polynomials $T_k(x)$ to the k^{th} order. The modified convolution is then defined as $g_\theta * x \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{L})x$ where $\theta \in \mathbb{R}^k$ and scaled Laplacian $\tilde{L} = 2L/\lambda_{max} - I_n$. This approximation results in spectral filters which are k-localized i.e. they depend on the k-hop neighborhood of the node. Kipf and Welling [17] further

restricted the convolution to single-hop i.e. immediate neighborhood of the node. The resulting convolution is linear in L and now has only two filter parameters, θ_0 and θ_1 which are shared over the whole graph.

$$g_\theta * x \approx \theta_0 x + \theta_1 (L - I_n) x$$

This can be approximated to $\theta_0 x - \theta_1 D^{-1/2} A D^{-1/2} x$. The authors further apply the following reduction $\theta = \theta_0 = -\theta_1$. reducing the above equation to the following additive form,

$$g_\theta \circledast x \approx \theta \left(I_n + D^{-1/2} A D^{-1/2} \right) x, \tag{1}$$

Given the input feature matrix X, each hidden layer of the convolutional network convolves the results of the previous layer,

$$Z^{(k)} = \sigma \left(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{1/2} Z^{(k-1)} W^{(k)} \right) \tag{2}$$

$Z^{(k)}$ is the output of the k^{th} layer, and $Z^{(0)} = X$. $\tilde{A} = A + I_N$ is the adjacency matrix (with self-loops) and $\tilde{D}$ the degree matrix. $W^{(k)}$ are filter θ paramters learnt by the network; $\sigma(.)$ denotes the activation function (e.g. ReLU, tanh).

The local graph convolution induces message-passing, with each node accumulating signals from its first-order neighborhood. In the k^{th} convolutional layer, $k \in [1, \ldots, K]$, the resulting signals incorporate the k-hop neighborhood of each node. In each layer, the convolution $\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{1/2}$ in eq. (2) results in each node's being represented by the average of it's own features and it's neighborhood, followed by the linear transform $W^{(k)}$. This implicitly encodes label and feature sharing across connected nodes with edges assumed to denote similarity or homogeneity between nodes.

We now revisit the contrastive relational view, having established the message-passing nature of eq. (2). In the contrastive view, we aim to discriminate adjacent nodes and invert the assigned labels, i.e. capture heterogeneity in neighborhoods rather than similarity. In this setting, eq. (2) fails to identify the contrast between the feature representations of accepted answer nodes and the rest. This results in misclassification of the accepted answers to the majority class of their neighborhoods. This effect is more prominent in larger cliques (questions with many answers) where the neighborhood shifts away from the accepted class of answers. While there have been extensions of graph convolution to the inductive setting [13], multi-relational setting [20] and signed networks [6], none of these approaches are suitable to model data induced contrastive views. In the next subsection, we propose modified convolution approach and show that it results in a form of *Discriminative Feature Magnification*, highlighting feature differences between accepted answers and their neighborhoods in the contrastive views.

4.2 Contrastive Convolution

The contrastive relational view encodes contrast of each node to its neighborhood, a complementary hypothesis to the label sharing effect described in the previous section. We revisit a key simplifying assumption in [17], $\theta = \theta_0 = -\theta_1$. This implictly encodes the similarity assumption in feature transformation. To establish contrast, we need to compute the *difference* instead of addition between node's own features I_n and neighborhood averaged features $D^{-\frac{1}{2}} A_c D^{\frac{1}{2}}$ in the clique. We can encode this contrastive relationship

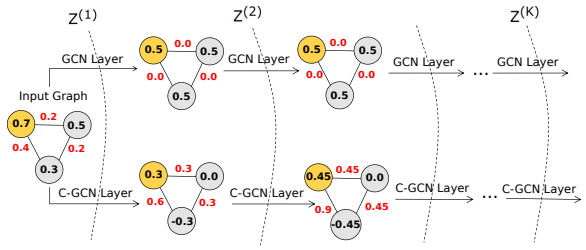


Figure 3: Stylized example showing the convolution results of GCN and proposed Contrastive GCN for a question with three answers. The feature difference between neighboring nodes increases with each convolution layer for Contrastive GCN while GCN averages the feature values among nodes.

by using same sign filter $\theta = \theta_0 = \theta_1$ which leads to this modified layer-wise convolution operation,

$$g_\theta * x \approx \theta \left(I_n - D^{-1/2} A D^{-1/2} \right) x, \tag{3}$$

and the output of the k^{th} convolutional layer is now given by,

$$Z_c^{(k)} = \sigma \left((I_n - D^{-1/2} A_c D^{1/2}) Z_c^{(k-1)} W_c^{(k)} \right) \tag{4}$$

with A_c denoting the adjacency matrix in the contrastive relational view (subscript c denotes the contrastive view). Note the effect of the change, each layer now computes the feature differences between each node and the local neighborhood, resulting in feature discrimination between contrasting nodes. Now we visit the term $(I_n - D^{-1/2} A_c D^{1/2}) Z_c^{(k-1)}$. Under this formulation, the convolved feature value for node i, with neighborhood set $\mathbb{N}_i$ is given by,

$$z_c^{(k)}(i) = \sigma \left(\left(z_c^{(k-1)}(i) - \frac{1}{|\mathbb{N}_i|} \sum_{j \in \mathbb{N}_i} z_c^{(k-1)}(j) \right) W_c^k \right) \tag{5}$$

Now consider a pair of contrasting nodes, i_1 and i_2 in the immediate neighborhood of each other. Note that in our tournament-style graph construction, the nodes within each question have identical degree (say $d = |A_q|$). Let us ignore the linear transform by setting $W_c^k = \mathbb{I}$ and set $\sigma(.)$ to the identity function. It is then easy to verify,

$$z_c^{(k)}(i_1) - z_c^{(k)}(i_2) = (1 + \frac{1}{d}) \times (z_c^{(k-1)}(i_1) - z_c^{(k-1)}(i_2)) \tag{6}$$

where $z_c^{(k)}(i)$ denotes the output of the k^{th} convolution layer for the i^{th} node in the contrastive view. As a result, each convolutional layer magnifies the feature contrast between the neighboring nodes, i.e. the contrasting nodes move futher apart. This achieves the desired *Discriminative Feature Magnification* effect. An illustration is provided in fig. 3 with a uni-dimensional feature. Contrasting nodes are shifted further apart by eq. (4) improving their separability in the learned manifold (further discussion in section 6.7).

Note that induced contrast relation is distinct from signed graphs where edges are either positive or negative. Signed graph convolution [6] gathers the friends and enemies of a given node and computes separate embeddings through feature sharing. This does not achieve node feature contrast. Also, signed edges are harvested

from curated data, unlike our induced links. While graph attention [27] could learn negative attention over neighbors in theory, it would involve significant computational expenses on large graphs.

4.3 Relative Similarity Convolution

We next discuss how to encode relative similarity relationships in the convolution framework. Recall from our discussion in section 3, a similarity edge connects two answers of a player if contrast with the skill of other players answering the same question is identical. As similarity relation enforce smoothness or sharing of labels among nodes, we can directly apply the vanilla formulation (Equation 2). Thus, output of the k-th convolutional layer for the similarity view, $Z_s^{(k)}$ is defined as:

$$Z_s^k = \sigma\left((I_N + D^{-\frac{1}{2}} A_s D^{\frac{1}{2}}) Z_s^{(k-1)} W_s^{(k)}\right) \tag{7}$$

In the previous section, we operationalized relative similarity with the TrueSkill measure and Arrival Similarity. The convolution operation is defined in the same way (as above) for both views. We refer to the representations as Z_{as}^k, Z_{ts}^k respectively.

4.4 Reflexive Convolution

Reflexive relationship is encoded with self-loops in the graph as they assume independence among nodes. This leads to an identity adjacency matrix and output of the k-th convolutional layer for the reflexive view, $Z_r^{(k)}$ reduces to:

$$Z_r^k = \sigma\left(I_n Z_r^{(k-1)} W_r^{(k)}\right) \tag{8}$$

Hence, reflexive convolution operation is equivalent to a feedforward neural network with multiple layers and activation $\sigma(.)$.

Weight Sharing: For multiple operationalizations of a class (e.g. TrueSkill and Arrival Similarity), we train a seperate GCN for each graph but share the layerwise linear-transforms $W_s^{(k)}$ to capture similarities in the learned latent spaces. In addition, we introduce an alignment loss term, which attempts to align the learned embedding representations (the loss term aligns pairs of embedding representations, $\sum_i ||Z_1^k(i) - Z_2^k(i)||$. In principle, multiple GCNs can augment each view by sharing prior knowledge through weight sharing and alignment loss (a similar idea was explored to capture local and global views in [34]). We provide the details of our training algorithms in section 5.

In this section, we propsed Graph Convolutional Networks as a node classification approach in our context. We introduced a simple alternate formulation to tackle the contrastive view and achieve discriminative magnification to improve class separability. We refer to the operations on each view as an Induced Relational Graph Convolutional Network (**IR-GCN**). We now proceed to formulate aggregation strategies to combine the representations learned by multiple IR-GCNs across the semantic classes of relational views.

5 AGGREGATING RELATIONAL VIEWS

In the previous section, we introduced three broad classes of induced relational views of the (q, a) tuples — Contrastive, Relative Similarity and Reflexive views to study their relational properties. We distinguish the relational class and a specific view such as TrueSkill, which is an instance of the Similarity class. Each relational class is expected to capture a semantically diverse neighborhoods of nodes. The convolution operator aggregates the neighbor information under each view. The key question that follows is, *how do we combine these diverse views in a unified learning framework?* In past work, multiple solutions have been considered:

- **Neighborhood Aggregation**: In this approach, we represent nodes by aggregating feature representations of it's neighbors across all views [13, 20].
- **Stacking**: Multiple convolution layers stacked end-to-end (each potentially handling a different view) [32].
- **Fusion**: Follows a multi-modal fusion approach [8], where views are considered distinct data modalities.
- **Shared Latent Structure**: Attempts to transfer knowledge across relational views (modalities) with constraints on the representations (e.g. [34] aligns embeddings across views).

Ensemble methods introduced in [20] work on multiple relational edges in knowledge graphs. None of these approaches are directly suitable for our induced relationships. Our relational classes carry different semantics (contrastive vs similarity vs reflexive) and aggregating them by pooling, concatenation or addition of embeddings fails to capture semantic heterogeneity of the induced views. Further, data induced relations are uncurated and inherently noisy. Directly aggregating the learned representations via Stacking or Fusion can lead to noise propagation.

In our contrastive view, we must achieve feature discrimination and label inversion between contrasting nodes, as opposed to label homogeneity and feature sharing in the similarity views. Gradient boosting techniques are known to improve performance when individual classifiers, including neural networks [21], are diverse yet accurate. A natural solution then is to apply boosting to the set of IR-GCNs and bridge the weakness of each learner. However, a direct application across all views is sub-optimal as it does not exploit the commonalities within a single relational class (e.g. TrueSkill and Arrival within Relative Similarity).

Consider the most general ontology of the chosen relations, namely a set of relational classes (such as Contrastive and Similarity), each with multiple views (e.g. True Skill, Arrival). We propose the following strategy to aggregate IR-GCNs across views.

Inter-class Aggregation: We expect distinct relational classes to perform well on different subsets of the set of (q, a) tuples. We empirically verify this with the Jaccard overlap between the set of misclassified nodes under each relational class on our datasets. Given $\mathbf{M}_A$ and $\mathbf{M}_B$, the sets of (q, a) tuples misclassified by IR-GCNs A and B respectively, the jaccard overlap is,

$$\mathcal{J}_{A,B} = \frac{\mathbf{M}_A \cap \mathbf{M}_B}{\mathbf{M}_A \cup \mathbf{M}_B}$$

The $\mathcal{J}_{A,B}$ values are as follows for the relational pairings: (Contrastive, TrueSkill Similarity) = 0.42, (Contrastive, Reflexive) = 0.44 and (Reflexive, TrueSkill Similarity) = 0.48. Relatively low values of the overlap metric indicate uncorrelated errors across the views. Gradient boosting methods can effectively aggregate relational class level representations, but are not optimal within a class (since it cannot capture shared latent structure between views in a class).

Intra-class Aggregation: Intra-class learning (with a single class like Similarity) can benefit from sharing the commonalities of views. In our setting we found True Skill and Arrival Similarity

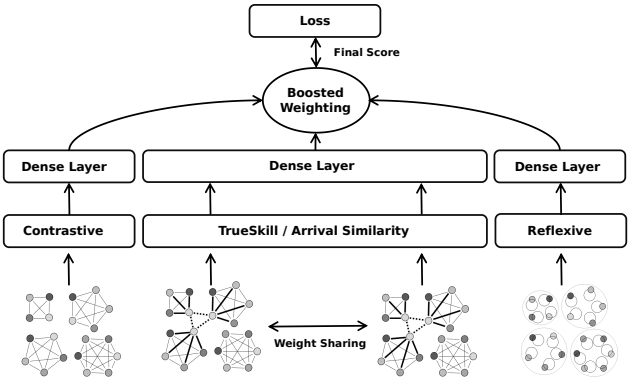


Figure 4: Schematic diagram of our proposed Boosted Induced-Relational GCN model.

GCNs to commit similar mistakes ($\mathcal{J}_{TS,AS}$ = 0.66). Sharing the structure of the their latent spaces can capture these similarities.

Motivated by the regularization term in [34], we follow a similar strategy to combine our similiarity view IR-GCNs. Now consider a relational class C with views $v \in C$ (e.g. TrueSkill, Arrival in the Similarity class). To share the latent structure of their representations, the following regularizer is introduced (t_i denotes a single (q, a) tuple),

$$\sum_i \sum_{(v,v') \in C} ||Z_v^K(t_i) - Z_{v'}^K(t_i)|| \tag{9}$$

The embedding representations are constrained to be similar at the last layer (k=K) for all pairs of views in a class.

In the previous sections we described three classes of relational views, Contrastive (c), Similarity (s) and Reflexive (r). For each class $C \in \{c, s, r\}$, we have a set of views $v \in C$ and each IR-GCN$_v$ generates embedding representation $Z_v^K(t_i)$. These Z_v^K's are jointly projected to generate a single relational class level score, h_C which is viewed as the prediction outcome for relational class C (eq. (10)). These class level outputs are then scaled and combined by the respective gradient boosting methods (we employed Adaboost [10] in our experiments). Figure 4 exhibits the overall architecture of our Boosted IR-GCN model.

$$h_C(t_i) = \sum_{v \in C} Z_v^K(t_i) * \tilde{W}_v \tag{10}$$

with transform parameters $\tilde{W}_v \in \mathbb{R}^{D \times 1}$ and output embedding (Z_v^K) dimensionality D.

Now we describe the boosted weights for the outputs of each relational class for each tuple t_i, namely $h_C(t_i)$. The weights $w_C(t_i)$ of the t_i^{th} tuple for the C^{th} relational class are given by $w_C(t_i) = e^{-y_i \times h_{(1\ldots C-1)}(t_i)}$, where $h_{(1\ldots C-1)}$ denotes the prediction output of the boosted classifier combining all relational classes before class C. In our answer selection problem, the contrastive relational class performs the best, and is hence added first to the ensemble, followed by relative similarity and relexive classes, i.e. $\{c, s, r\}$. The above description is formalized in algorithm 1.

Training Algorithm: We now describe the loss function and the training algorithm for our Boosted IR-GCN model described in Algorithm 2. For each epoch, we first compute the aggregated prediction score $h_{(1\ldots C)}$ of our boosted model as described in algorithm 1. We use exponential loss L_b to learn the weight parameters and apply elastic-net regularization to the convolutional weight matrices W_c^k, W_s^k, W_r^k. Note that we employ weight sharing between all views sharing the same relational class so that only one set of weight matrices is learned per relational class. The views in our training are Tournament-Style Contrast (c), TrueSkill (ts), Arrival Similarity (as) and Reflexive view (r). Note that the reg-

Algorithm 1 Adaboost IR-GCN Forward

function BOOSTEDMODEL(X, Y, A)
  for $C \in \{c, s, r\}$ **do**
    $h_C(t_i) = \sum_{v \in C} Z_v^K(t_i) * \tilde{W}_v; \forall v \in C$ ▷ Equation 10
    $w_C(t_i) \leftarrow \exp^{-y_i * h_{(1\ldots C-1)}(t_i)}$
    $\alpha_C \leftarrow \frac{1}{2} \ln \frac{\sum_{y_i = h_C(t_i)} w_C(t_i)}{\sum_{y_i \neq h_C(t_i)} w_C(t_i)}$
    $h_{(1,\ldots,C)}(t_i) \leftarrow h_{(1\ldots C-1)}(t_i) + \alpha_C * h_C(t_i)$
    ▷ Update boosted GCN
  end for
  return $h_{(1,\ldots,C)(t_i)}$ ▷ Final Aggregated Prediction
  return $Z_v^K(t_i) \ \forall \ v \in \{c, s, r\}$ ▷ IR-GCN Embeddings
end function

Algorithm 2 Adaboost IR-GCN Training

Input: Tuples $t_1 \ldots t_n$, Class labels $y_1 \ldots y_n$,$y_i \in \{-1, 1\}$, Adjacency matrix of each IR-GCN A_c, A_{ts}, A_{as}, A_r

Output: Trained Model i.e. Weight parameters $W_m^{(1)} \ldots W_m^{(k)} \forall m \in \{c, s, r\}$ and transform parameters $\tilde{W}_v$, $v \in \{c, ts, as, r\}$

while *not converged* **do**
  $h_b(t_i), Z_v^K(t_i) \leftarrow BoostedModel(X, Y, A)$ ▷ Algorithm 1
  $\mathcal{L}_b \leftarrow \sum_{i=1}^{n} \exp^{-y_i * h_b(x_i)} + \lambda_1 \mathcal{L}_1(.) + \lambda_2 \mathcal{L}_2(.)$
  if t % 3 == 0 **then**
    $\mathcal{L}_{as} \leftarrow \sum_{i=1}^{n} \exp^{-y_i * Z_{as}^K(t_i) \tilde{W}_{as}}$
    $\mathcal{L}_{ts} \leftarrow \sum_{i=1}^{n} \exp^{-y_i * Z_{ts}^K(t_i) \tilde{W}_{ts}}$
    $\mathcal{L}_s \leftarrow \mathcal{L}_{as} + \mathcal{L}_{ts} + ||Z_{as} - Z_{ts}||$
    $\mathcal{L}_b \leftarrow \mathcal{L}_b + \lambda(t) \mathcal{L}_s$
  end if
  if t % 3 == 1 **then**
    $\mathcal{L}_r \leftarrow \sum_{i=1}^{n} \exp^{-y_i * Z_r^K(t_i) \tilde{W}_r}$
    $\mathcal{L}_b \leftarrow \mathcal{L}_b + \lambda(t) \mathcal{L}_r$
  end if
  if t % 3 == 2 **then**
    $\mathcal{L}_c \leftarrow \sum_{i=1}^{n} \exp^{-y_i * Z_c^K(t_i) \tilde{W}_c}$
    $\mathcal{L}_b \leftarrow \mathcal{L}_b + \lambda(t) \mathcal{L}_c$
  end if
  Update $W_c^{(k)}, W_s^{(k)}, W_r^{(k)}$ using $\frac{\partial \mathcal{L}_b}{\partial W_c^{(k)}}, \frac{\partial \mathcal{L}_b}{\partial W_s^{(k)}}, \frac{\partial \mathcal{L}_b}{\partial W_r^{(k)}}$
  Update transform parameters $\tilde{W}_{as}, \tilde{W}_{ts}, \tilde{W}_r, \tilde{W}_c$
end while

ularizer $||Z_{as} - Z_{ts}||$ ensures close coupling between the latent representations learned between views in the same relational class. For instance, we want to facilitate information sharing between the TrueSkill similarity and Arrival similarity relations. Thus, if an answer is authored by a user with a higher skill rating and answered significantly earlier than other answers, its probability to be credible should be mutually enhanced by both signals. We also share the weight matrices $W_s^k \forall k \in [1, \ldots, K]$ to enable analogous data transformations between TrueSkill and Arrival Similarity.

IR-GCN Annealing: We apply an exponential annealing schedule, $\lambda(t)$ to the loss functions of each IR-GCN. As training progresses and the boosted model learns to optimally distribute nodes among the relational classes, increase in $\lambda(t)$ ensures more emphasis is provided to the individual convolutional networks.

6 EXPERIMENTS

In this section, we first provide data descriptions followed by our experimental setup; comparative baselines, evaluation metrics and implementation details. We then present results across diverse experiments to evaluate the performance of our model on merging semantically diverse induced-relations. We finally provide a brief analysis of our proposed contrastive modification to convolution, gradient boosting and their implications on model fitting.

6.1 Dataset

We evaluate our approach on two large online Community Question Answer (CQA) paltforms, Stack Exchange and Reddit with multiple datasets from each platform. Table 1 shows the dataset statistics.

StackExchange[1] is a popular Community Question Answer platform with communities catering to topics ranging from open-ended discussions to fact-based questions. For our anlaysis, we collect data from five tech-based (arduino, bitcoin, unix, crypto and askUbuntu) and five non-tech communities (movie, history, parenting, music and photography) until August 2018. In StackExchange, each questioner can accept an answer out of candidate answers. We only consider questions with an accepted answer.

For each (q, a) tuple, we compute following basic features:
Activity features : View count of the question, number of comments for both question and answer, difference between posting time of question and answer, arrival rank of answer (we assign rank 1 to the first posted answer) [26].
Text features : Paragraph and word count of question and answer, presence of code snippet in question and answer (useful for programming based forums), word count in question title.
User features : Word count in user profile's Aboutme section for both questioner and answerer.

Time dependent features like upvotes/downvotes of the post and user features like reputation or badges used in earlier studies on StackExchange [1] are problematic for two reasons. First, we only know the aggregate values not how these values change with time. Second since these values typically increase over time, it is unclear if an accepted answer received the votes *prior* to or *after* an answer was accepted.

[1]https://stackexchange.com/

Reddit[2] is another popular CQA platform with subreddits similar to StackExchange communities. In particular, we focus on Ask* subreddits as they are primarily used to seek help from a community of experts and non-experts. In particular, we crawled data from /r/askscience (science forum), /r/AskHistorians (history forum), and /r/AskDocs (medical forum) until October 2017. We performed basic preprocessing and removed posts or comments with single word/URLs or missing author/title information. We also removed infrequent users who posted less than two comments. Reddit has a hierarchical comment structure. For this paper, we treat first-level comments as potential answers to the question. Users in these subreddits can get verified by providing anonymized verification documents including certification numbers, contact information etc. to the moderators. We denote these verified users as experts. We treat an expert's comment as equivalent to an accepted answer and only consider posts which have an expert answer for our experiments. We discard posts with multiple experts' comment as it is hard to objectively choose a winner.

We employ 12 basic features for the Reddit dataset:
Activity features : ArrivalRank of the answer, Number of subsequent comments on the answer, Number of other answers to the question, Upvotes and downvotes for both, question and answer.
Text features : Word count of the question and answer

We employ post vote features here as [11] showed that there is widespread under-provision of voting on Reddit, partially due to long comment threads. It can act as a weak signal for answer quality. Unlike the StackExchanges, Reddit voting is not biased by publicly visible acceptance of answers to a question. Thus, votes ideally represent independent judgement of the crowd.

6.2 Experimental Setup

6.2.1 Baselines. We compare against state-of-the art feature-based baselines for answer selection and competing aggregation strategies to fuse diverse relational views of the dataset [20, 34].

Random Forest (RF) [1, 26] model is trained on the feature set mentioned earlier for each dataset. This model has been shown to be the most effective feature-based model for Answer Selection.

Feed-Forward network (FF) [15] is used as deep learning baseline to learn non-linear transformations of the feature vectors for each (q, a) tuple. This is equivalent to our Reflexive GCN model in isolation.

Dual GCN (DualGCN) [34] In addition to the supervised loss computed using training labels, they introduce a regularizer to minimize mean squared error (MSE) between two GCN's representations, thus aligning the learned latent spaces. [34] proposed the model for two GCN representations. We extend it to four IR-GCNs representing our relational views between the nodes. We minimize the direct loss term for the best performing IR-GCN model and it's latent structure alignment with respect to all other IR-GCN's representations. The Contrastive view is seen to exhibit the best performance in isolation. Thus, the DualGCN loss can be given by:

$$\mathcal{L} = \mathcal{L}_0(Z_c) + \lambda(t) \left(\mathcal{L}_{reg}(Z_c, Z_{ts}) + \mathcal{L}_{reg}(Z_c, Z_{as}) + \mathcal{L}_{reg}(Z_c, Z_r) \right)$$

Relational GCN (RelationalGCN) [20] combines the output representations $Z_v^{(k-1)}$ of layer $k-1$ of each IR-GCN model to

[2]https://www.reddit.com/

Non-Tech StackExchange	Q	A	U	$\mu_a(q)$	Tech StackExchange	Q	A	U	$\mu_a(q)$	Reddit	Q	A	U	$\mu_a(q)$
movie	3,876	11,545	4,891	2.98	arduino	2,483	6,301	1,751	2.54	AskDocs	11,189	29,207	4,530	2.61
history	2,457	8,074	2,080	3.27	bitcoin	3,057	8,280	2,657	2.71	AskHistorians	15,425	45,586	11,761	2.96
parenting	1,790	7,744	2,458	4.33	unix	9,207	33,980	7,987	3.69	AskScience	37,990	121,278	32,117	3.19
music	4,497	15,763	4,054	3.51	crypto	2,893	7,132	1,530	2.47	-	-	-	-	-
photo	7,626	26,768	4,931	3.50	askUbuntu	41,192	119,248	41,947	2.89	-	-	-	-	-

Table 1: Dataset statistics for the Stack Exchange and Reddit datasets. Q: number of questions; A: number of answers; U: number of users; $\mu_a(q)$: mean number of answers per question. Non-Tech StackExchange have slightly more answers per question than Tech StackExchange. Ask* Reddits are the largest dataset in our dataset.

compute an aggregated input to layer k. Formally,

$$Z_{rgcn}^{(k)} = \sigma\left(\sum_{v \in \{c,ts,as,r\}} Z_v^{(k-1)}\right)$$

where Z_{rgcn} is final output of this model at layer k and σ is the activation function.

We also report results for each IR-GCN: Contrastive (C-GCN), Arrival Similarity (AS-GCN), TrueSkill Similarity (TS-GCN) and Reflexive (R-GCN) with our proposed Boosted IR-GCN model. We do not compare with other structure-based approaches to compute node representations [12, 19, 24, 31] as GCN is shown to outperform them [17]. We compare with common aggregation strategies to merge neural representations in section 6.5.

6.2.2 Evaluation Metric. We evaluate our model on two metrics, Accuracy and Mean Reciprocal Rank (MRR). Accuracy metric is widely used in node classification literature while MRR is popular for ranking problems like answer selection. Formally, $Acc = n^{-1}\sum_{i=1}^{n} \mathbb{1}(y_i, h_b(x_i))$ with $\mathbb{1}(.)$ is the indicator function and $MRR = \sum_{q=1}^{Q} (R_a{}^{-1})\ \forall a \in A(q)$ and $y_a = 1$ where R_a refers to the position of accepted answer for q-th question among ranked list of answers in decreasing order of their model output score [29].

6.2.3 Implementation Details. We use four hidden layers in each induced-relational GCN with hidden dimensions 50, 10, 10, 5 respectively and ReLU activation. We use 50% dropout with ADAM optimizer [16] for training. We set coefficients of L_1 and L_2 regularizers to $\lambda_1 = 0.05$ and $\lambda_2 = 0.01$ respectively. For TrueSkill Similarity, we use margin $\delta = 4$ to create links, while for Arrival similarity, we use $\gamma = 0.95$. We implement a mini-batch version of GCN with one question per batch for large graphs. This is equivalent to training on whole graph as we have disconnected cliques. All models are implemented in Pytorch and code will be released upon publication.

6.3 Performance Analysis

Table 2 and 3 shows impressive gains over the state-of-art baselines for both the datasets. All results are reported after 5-fold cross validation. Our boosted induced-relational GCN model beats best performing baseline by 4-5% on average in accuracy for both Tech and Non-Tech StackExchange. For Reddit, the margin is much higher. Our model improves by 16% over the baselines. The improvement in MRR values for both Tech and Non-Tech StackExchange are around 1.5-2%. The improvement in MRR is again higher for Reddit at an average increase of 7% than the baseline. Note that MRR is based only on the rank of accepted answer while accuracy is based on correct labeling of *both* accepted and non-accepted answers.

Among individual induced-relational GCNs, for StackExchange, Contrastive performs best closely followed by Arrival Similarity and TrueSkill Similarity. While for Reddit, there is huge difference in performance for each induced-relational GCN. TrueSkill Similarity performs much better followed by Arrival Similarity and Contrastive. Reflexive GCN performs the worst for both datasets as it predicts each node's label independent of answers to the same question.

Out of the baseline graph ensemble approaches, DualGCN and RelationalGCN, DualGCN consistently performs better than RelationalGCN by an average of around 2.5% and 3% for StackExchange and Reddit respectively.

Recall that in RelationalGCN model, the convolution output of each IR-GCN is linearly combined to compute the final convolution output. This works well for knowledge graphs as each relational class can be thought of as a feature and then it accumulates information from each feature. DualGCN is similar to our approach and trains different GCN for each relation and later merges their results. However, it enforces similarity in node representations learned by each induced-relation GCN. This is not suitable for our induced-relationships as they are semantically different (contrastive captures contrast in features vs similarity enforces label sharing).

6.4 Ablation Study on Relational Views

We present results of ablation study with different combination of relational views (Contrastive, Similarity and Reflexive) used for Boosted IR-GCN model in Table 4. Similarity (TrueSkill and Arrival) relation used in isolation perform the worst among all variants in stackexchange. Training Contrastive and Similarity relation together in our boosted framework performs similar to our final proposed model. Reflexive GCN contributes the least which is expected as it doesn't take any neighbors into consideration.

6.5 Aggregator Architecture Variants

We compare our gradient boosting based aggregation approach with other popular methods used in literature to merge different neural networks discussed in section 5.

Table 5 reports the accuracy results for these aggregator variants as compared to our model. Our method outperforms all the variants with Fusion performing the best among the baselines. Note that for StackExchange, these approaches perform worse than just strongest IR-GCN, Contrastive GCN. This reaffirms that existing aggregation models are not suitable for our problem.

Method	Non-Tech StackExchanges										Tech StackExchanges									
	movie		history		parenting		music		photo		arduino		bitcoin		unix		crypto		askUbuntu	
	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR
RF [1, 26]	70.91	0.704	72.83	0.683	75.04	0.507	70.33	0.628	69.27	0.580	63.74	0.717	65.05	0.701	70.90	0.597	63.13	0.729	64.63	0.667
FF [15]	72.75	0.824	72.69	0.806	76.97	0.717	76.07	0.773	73.66	0.747	65.93	0.802	68.18	0.803	73.14	0.730	68.09	0.801	68.49	0.789
DGCN [34]	77.62	0.839	76.73	0.810	76.79	0.748	76.81	0.785	75.78	0.752	71.14	0.810	70.32	0.806	75.16	0.749	71.34	0.809	70.25	0.794
RGCN [20]	61.97	0.649	63.21	0.661	62.97	0.578	53.95	0.614	56.12	0.624	60.86	0.696	64.60	0.691	64.14	0.610	51.00	0.704	55.24	0.636
AS-GCN	74.02	0.836	74.14	0.805	78.56	0.720	76.62	0.790	74.60	0.751	66.91	0.791	68.95	0.795	74.90	0.749	69.63	0.800	70.15	0.774
TS-GCN	72.40	0.831	74.08	0.791	77.96	0.704	76.96	0.784	74.19	0.727	65.67	0.789	68.46	0.799	73.95	0.748	69.11	0.805	69.49	0.782
C-GCN	78.25	0.843	76.90	0.815	78.91	0.751	77.23	0.794	76.88	0.757	69.48	0.811	72.70	0.810	76.44	0.761	73.09	0.812	72.27	0.792
B-IR-GCN	**80.27**	**0.850**	**80.22**	**0.821**	**81.61**	**0.771**	**79.39**	**0.808**	**78.26**	**0.770**	**72.40**	**0.816**	**74.61**	**0.824**	**78.75**	**0.765**	**74.29**	**0.821**	**75.25**	**0.798**

* DGCN stands for DualGCN, RGCN stands for RelationalGCN, and B-IRGCN stands for Boosted IR-GCN.

Table 2: Accuracy and MRR values for Stackexchange with state-of-the-art baselines. Our model outperforms by at least 4% in Accuracy and 1.5% in MRR. Contrastive GCN performs best among individual IR-GCNs.

Method	AskDocs		AskHistorians		AskScience	
	Acc (%)	MRR	Acc(%)	MRR	Acc(%)	MRR
RF [1, 26]	59.35	0.698	65.62	0.709	65.87	0.706
FF [15]	62.30	0.715	67.89	0.730	68.99	0.713
DualGCN [34]	77.54	0.790	80.49	0.805	75.57	0.821
RelationalGCN [20]	57.98	0.667	64.56	0.684	62.42	0.642
AS-GCN	76.53	0.794	80.70	0.781	78.14	0.797
TS-GCN	84.44	0.861	90.95	0.829	87.61	0.822
C-GCN	67.39	0.753	70.57	0.744	71.11	0.769
Boosted IR-GCN	**87.60**	**0.896**	**93.81**	**0.851**	**89.11**	**0.837**

Table 3: Accuracy and MRR values for Ask Reddits. Our model significantly outperforms by 16% in Accuracy and 7% in MRR. TrueSkill Similarity performs best among individual IR-GCNs.

IR-GCN	movie	unix	AskDocs
{ TS, AS }	75.46	74.90	84.57
{TS, AS } + R	76.47	74.99	86.34
C + R	76.02	77.53	70.02
C + { TS, AS }	79.88	78.02	86.99
C + { TS, AS } + R	**80.27**	**78.75**	**87.60**

Table 4: 5-fold Accuracy (in %) comparison for different combination of relational classes for our boosted model. Contrastive and Similarity are most important relational classes.

Method	movie	unix	AskDocs
Stacking [32]	75.82	74.59	85.40
Fusion [8]	78.60	74.82	86.33
NeighborAgg [13, 20]	76.02	75.84	86.00
Boosted IR-GCN	**80.27**	**78.75**	**87.60**

Table 5: 5-fold Accuracy (in %) comparison of different aggregator architectures. These architectures perform worse than Contrastive GCN for StackExchange.

6.6 Label Sparsity

Graph Convolution Networks are robust to label sparsity as they exploit graph structure and are thus heavily used for semi-supervised settings. Figure 5 shows change in accuracy for movie StackExchange at varying training label rate. Even though our graph contains disconnected cliques, Boosted IR-GCN still preserves robustness to label sparsity. In contrast, accuracy of feed forward model

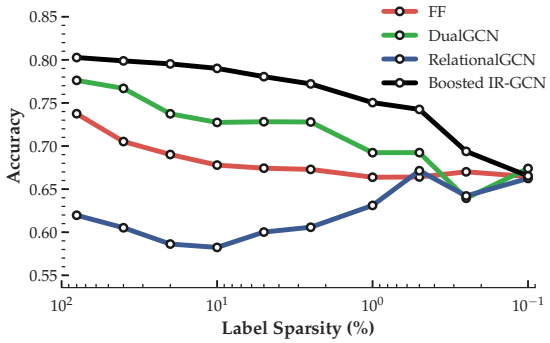


Figure 5: Change in accuracy with varying training label rates for movie StackExchange. Our model is more robust to label sparsity than other relation ensemble approaches.

declines sharply with less label information. Performance of DualGCN falls sharply with decrease in label rate while RelationGCN's performance improves with decrease in label rate. In case of extremely low label rate of 0.1%, all approaches converge to the same value, which is the expectation of theoretically random selection.

6.7 Contrastive GCN Analysis

The ability of neural networks to perform classification in sparse high-dimensional manifolds has been studied in past work, especially in the context of adversarial learning [18]. We employ the ReLU activation function in our convolution layers and study the outputs of the kth layer, i.e. embeddings with k-order locality. This transformation breaks the input space into cells with smooth gradients within each cell, at whose boundaries the piecewise linear function changes (i.e. the likelihood of the two classes of answers).

We ask a specific question in the context of our Contrastive IR-GCN. *What is the impact of the layerwise discriminative magnification induced by our formulation?* Disciminative magnifications results in improved separability of the two classes in the later convolving layers, an effect we earlier demonstrated with a sample network in fig. 3. This positively impacts the ability of the model to explain the observed data points (i.e create p-domains that are well aligned with the contrastive samples provided) and improve the generalizability of the learned model to unseen data points. However, it is important to maintain sufficient regularization with

weight decay to prevent sparse regions exhibiting sharp gradients which could affect model performance.

The capacity of our model can also be quantified in terms of the VC dimension of the aggregated classifier against the individual learners. Gradient boosting with multiple relation learners (each of which captures a specfic aspect of node locality via graph convolution on the induced relations) could boost the capacity of the joint model, enabling better generalization and a more accurate fit in the data manifold (i.e. higher capacity to fit regions to fine distinctions).

Let us denote the upper bound of the VC dimension or capacity of each individual learner as D (If the individual learners do not have identical capacity, the minimum can be used to compute a lower bound on the aggregated learner capacity). Then the gradient boosted learner with T classifiers has a bound on it's capacity [22] given by,

$$\mathcal{VC}_{Agg} = T \times (D+1) \times (3\log(T.(D+1)) + 2)$$

Thus we identify two potential reasons for our performance gains, first the discriminative magnification effect which also supports the strong individual performance of the contrast view, and second the gain in capacity from boosting, which could explain it's advantage over competing aggregation methods.

6.8 Limitations

We do recognize certain limitations of our work. First, creating induced links require domain knowledge or empirical analysis of the dataset. Second, we do not exploit content in our model. Although we achieve significant improvements with basic feature set, content could be helpful in semi-supervised settings. Third, we assume no evolution in author skills as our model assumes each tournament occurs simultaneously. This is not true as users evolve over time with experience. We aim to address this in future work.

Our model showed significant gains over state-of-the art baselines for combining information from semantially disparate relational links in a graph. Our model is also more robust to training label sparsity as compared to other aggregator GCN approaches. We reasoned that the performance gains achieved by our aggregation strategy could be attributed in part to the enhanced learning capacity of the boosted model and the effect of discriminative feature magnification. Finally, we presented a few limitations and potential future extensions.

7 RELATED WORK

Our work is related to two main research areas; Answer Selection and Graph Convolution.

Answer Selection In community question answering forum (CQA), the past few decades have witnessed plenty of works targeting on answer selection. Based on the proposed models, previous literatures can be divided into two main paradigms: feature-driven models and deep text models.

Feature-Driven Model [1, 15, 25, 26]: Feature-driven models [1] develop features from three different perspectives: 1) *User features* from questioner and answerer; 2) *Content features* from question and answer itself; 3) *Thread features* from the relationship within one answer pool. These features are fed into classifiers, such as tree based models [1, 15, 26] to identify the best answer. Tian et al. [26] found that the best answer is usually the earlier and most different one, and tends to have more details and comments. Jenders et al. [15] trained several classifiers for online MOOC forums. Different from existing works, Burel et al. [1] emphasize on the thread-like structure of Q&A, and introduce four thread-based normalization methods. These models predict answer label independently of the other answers for the question.

Deep Text Models [23, 28, 30, 33]: Text based deep learning models learn optimal representation of QA text pairs to select the best answer. Feng et al. [9] augment CNN with discontinuous convolution for a better vector representation; Wang and Nyberg [28] uses a stacked biLSTM to match the meaning of question and answer; Sukhbaatar et al. [23] use attention mechanism in an end-to-end memory framework. Text based models take longer to train and are computationally expensive.

Graph Convolution can be applied in both spatial and spectral domains to compute node representations. The learned node representations are then used for various tasks like node classification [17], link prediction [20] etc. Approaches working in spatial domain generally use random walk statistics or k-hop neighborhood information to compute node representations [12, 19, 24, 31]. Notable work on graph convolution in spectral domain are done by [5, 7] which use fast localized convolutions. Our work is inspired by Graph Convolution Networks (GCN) [17] which has shown to perform better than spatial convolutions and is scalable to large graphs. Various extensions to the GCN model have been proposed for signed networks [6], inductive settings [13] and multiple relations [20, 34]. All of these approaches assume label sharing which doesn't work well for our induced contrastive relations.

8 CONCLUSION

This paper addressed the question of identifying appropriate content in social network. The question is challenging since the "best" content is determined in relation to other content; the social interaction graph further influences the identification. We developed a novel data induced relational graph convolutional framework to address this question. We made three contributions. First, we introduced a novel idea of using abstract relationships—contrastive, similar, reflexive—to induce different graphs on content generated in a social network. Second, we showed how to operationalize each relation to a Graph Convolutional Network architecture. Finally, we show through extensive empirical results that boosting techniques improved learning in our convolutional model. This was a surprising result since much of the work on neural architecture focuses on stacking, fusion or aggregator architectures. Our ablation studies show that the contrastive relation is most effective individually in StackExchange datasets, with TrueSkill similarity relation working best in the case of Reddit. We conduct extensive experiments with excellent experimental results over the state of the art baselines on two different CQA forums—StackExchange and Reddit. As part of future work, we plan to fold into our model temporal evolution of skill of the players into the convolutional framework.

REFERENCES

[1] Grégoire Burel, Paul Mulholland, and Harith Alani. 2016. Structural Normalisation Methods for Improving Best Answer Identification in Question Answering

Communities. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume*. 673–678. https://doi.org/10.1145/2872518.2890570

[2] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *Proceedings of the 22Nd International Conference on Machine Learning (ICML '05)*. ACM, New York, NY, USA, 89–96. https://doi.org/10.1145/1102351.1102363

[3] Christopher J. Burges, Robert Ragno, and Quoc V. Le. 2007. Learning to Rank with Nonsmooth Cost Functions. In *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. C. Platt, and T. Hoffman (Eds.). MIT Press, 193–200. http://papers.nips.cc/paper/2971-learning-to-rank-with-nonsmooth-cost-functions.pdf

[4] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research. http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. 3837–3845. http://papers.nips.cc/paper/6081-convolutional-neural-networks-on-graphs-with-fast-localized-spectral-filtering

[6] Tyler Derr, Yao Ma, and Jiliang Tang. 2018. Signed Graph Convolutional Network. *CoRR* abs/1808.06354 (2018).

[7] David K. Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. 2224–2232. http://papers.nips.cc/paper/5954-convolutional-networks-on-graphs-for-learning-molecular-fingerprints

[8] Golnoosh Farnadi, Jie Tang, Martine De Cock, and Marie-Francine Moens. 2018. User Profiling Through Deep Multimodal Fusion. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 171–179. https://doi.org/10.1145/3159652.3159691

[9] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA, December 13-17, 2015*. 813–820. https://doi.org/10.1109/ASRU.2015.7404872

[10] Yoav Freund and Robert E. Schapire. 1995. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. In *Proceedings of the Second European Conference on Computational Learning Theory (EuroCOLT '95)*. Springer-Verlag, London, UK, UK, 23–37. http://dl.acm.org/citation.cfm?id=646943.712093

[11] Eric Gilbert. 2013. Widespread Underprovision on Reddit. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13)*. ACM, New York, NY, USA, 803–808. https://doi.org/10.1145/2441776.2441866

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 855–864. https://doi.org/10.1145/2939672.2939754

[13] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[14] Ralf Herbrich, Tom Minka, and Thore Graepel. 2006. TrueSkill™: A Bayesian Skill Rating System. In *Proceedings of the 19th International Conference on Neural Information Processing Systems (NIPS'06)*. MIT Press, Cambridge, MA, USA, 569–576. http://dl.acm.org/citation.cfm?id=2976456.2976528

[15] Maximilian Jenders, Ralf Krestel, and Felix Naumann. 2016. Which Answer is Best?: Predicting Accepted Answers in MOOC Forums. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume*. 679–684. https://doi.org/10.1145/2872518.2890567

[16] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). arXiv:1412.6980 http://arxiv.org/abs/1412.6980

[17] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR* abs/1609.02907 (2016). arXiv:1609.02907 http://arxiv.org/abs/1609.02907

[18] Jiajun Lu, Theerasit Issaranon, and David A Forsyth. 2017. SafetyNet: Detecting and Rejecting Adversarial Examples Robustly.. In *ICCV*. 446–454.

[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. 701–710. https://doi.org/10.1145/2623330.2623732

[20] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*. Springer, 593–607.

[21] Holger Schwenk and Yoshua Bengio. 2000. Boosting Neural Networks. *Neural Computation* 12, 8 (2000), 1869–1887. https://doi.org/10.1162/089976600300015178

[22] Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

[23] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-To-End Memory Networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. 2440–2448. http://papers.nips.cc/paper/5846-end-to-end-memory-networks

[24] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*. 1067–1077. https://doi.org/10.1145/2736277.2741093

[25] Qiongjie Tian and Baoxin Li. 2016. Weakly hierarchical lasso based learning to rank in best answer prediction. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2016, San Francisco, CA, USA, August 18-21, 2016*. 307–314. https://doi.org/10.1109/ASONAM.2016.7752250

[26] Qiongjie Tian, Peng Zhang, and Baoxin Li. 2013. Towards Predicting the Best Answers in Community-based Question-Answering Services. In *Proceedings of the Seventh International Conference on Weblogs and Social Media, ICWSM 2013, Cambridge, Massachusetts, USA, July 8-11, 2013*. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/view/6096

[27] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* 1, 2 (2017).

[28] Di Wang and Eric Nyberg. 2015. A Long Short-Term Memory Model for Answer Sentence Selection in Question Answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*. 707–712. http://aclweb.org/anthology/P/P15/P15-2116.pdf

[29] Xin-Jing Wang, Xudong Tu, Dan Feng, and Lei Zhang. 2009. Ranking Community Answers by Modeling Question-answer Relationships via Analogical Reasoning. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '09)*. ACM, New York, NY, USA, 179–186. https://doi.org/10.1145/1571941.1571974

[30] Wei Wu, Houfeng Wang, and Xu Sun. 2018. Question Condensing Networks for Answer Selection in Community Question Answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. 1746–1755. https://aclanthology.info/papers/P18-1162/p18-1162

[31] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. 40–48. http://jmlr.org/proceedings/papers/v48/yanga16.html

[32] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal Graph Convolutional Neural Network: A Deep Learning Framework for Traffic Forecasting. *CoRR* abs/1709.04875 (2017). arXiv:1709.04875 http://arxiv.org/abs/1709.04875

[33] Xiaodong Zhang, Sujian Li, Lei Sha, and Houfeng Wang. 2017. Attentive Interactive Neural Networks for Answer Selection in Community Question Answering. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*. 3525–3531. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14611

[34] Chenyi Zhuang and Qiang Ma. 2018. Dual Graph Convolutional Networks for Graph-Based Semi-Supervised Classification. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 499–508. https://doi.org/10.1145/3178876.3186116