Integrating of Hierarchical Taxonomy as a Prior Knowledge in Severely Imbalanced Text Classification

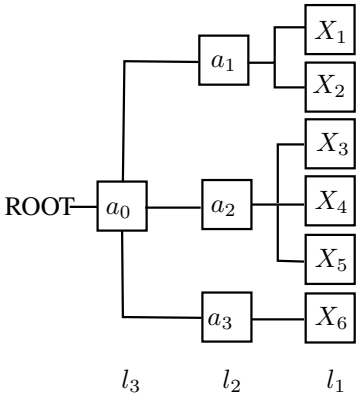
Abstract

In this paper we address the challenge of handling severe class imbalance. We investigate the effect of integrating hierarchical taxonomy of classes as a prior knowledge into the learning algorithm. We introduce two methods to integrate such the prior knowledge as an explicit regularizer into the loss function of learning algorithms. A Neural network, by reasoning on a hierarchical taxonomy, alleviates its output distributions over the classes to allow conditioning on upper concepts for a minority class. We limit ourself to text classification task and provide our experimental results on industrial real-world datasets. Our obtained results show a considerable improvement in performance of the trained models in fully supervised and semi-supervised fashions.

1 Introduction

Many real-world classification problems have an imbalanced class distribution, e.g., the occurrence of 25 fraudulent transactions among 1,000,000 normal transactions within a financial security dataset of a reputable bank (Hasanin et al. 2019). In current fully supervised classification tasks, a model is trained on a labeled dataset, in which mainly the labels are injected in the objective function (e.g., cross entropy) as a prior knowledge. Usually, this labels come from a bigger prior knowledge in a hierarchical taxonomy that allows a comprehensive reasoning over the labels. For example, in a classification task with 6 classes $\{$*screen_broken, Hardware_upgrade, Bios_update, Windows_installation, T-shirt, Sweater*$\}$ a higher level of concept would be $\{$*(screen_broken)* $->$ *Mobile_service, (Hardware_upgrade, Bios_update, Windows_installation)* $->$ *Computer_service, (T-shirt, Sweater)* $->$ *Clothing*$\}$.

In this paper, we aim to inject a hierarchical taxonomy of classes into the loss function of the learning algorithm of a text classification. We introduce two methods to represent and inject the hierarchical taxonomy. In the first method (Section 3.1), it is represented in form of a constraints in Boolean logic. For example, Figure 1 shows a hierarchical taxonomy for class labels, in which leaves in level l_1 indicate real class labels used in (e.g. cross entropy) loss function, and nodes in a higher level l_2 indicate a higher level of concept for the labels and usually don't use in classification algorithm. Our goal is to use higher level of the taxonomy to restrict distribution of a neural network outputs.



$$\alpha = (\neg X_1 \wedge \neg X_2 \wedge X_3 \wedge \neg X_4 \wedge \neg X_5 \wedge \neg X_6) \vee (\neg X_1 \wedge \neg X_2 \neg \wedge X_3 \wedge X_4 \wedge \neg X_5 \wedge \neg X_6) \vee (\neg X_1 \wedge \neg X_2 \neg \wedge X_3 \wedge \neg X_4 \wedge X_5 \wedge \neg X_6)$$

Figure 1: A symbolic representation/sentence for node a_2 in a higher level l_2 of a hierarchical taxonomy for multi-class classification

Similar to (Xu et al. 2018), we augment neural networks with the ability to learn how to make predictions subject to these constraints, and use the symbolic knowledge to improve the learning performance. In the second method (Section 3.2), we use Graph Convolutional Networks (GNN) to represent and inject the hierarchical taxonomy into the loss function. We compare the obtained results from these methods with and without integrating the hierarchical taxonomy of classes, explicitly, into the loss function of the learning algorithm. Our experimental results show the significant effect of how higher levels of hierarchical taxonomy can alleviate unequal distribution of classes in severe imbalanced classification.

Our contributions in this paper is in flat classification, referring to standard binary or multi-class classification problem, and it differs from hierarchical classification, where the class set to be predicted are organized into a class hierarchy–typically a tree or a DAG (Direct Acyclic Graph).

2 Related Work

Approaches in dealing with imbalanced classification problem can be grouped in three categories: data-level approaches, algorithm-level techniques, and hybrid methods (Johnson and Khoshgoftaar 2019). In data-level approaches, the goal is to alleviate unequal distribution of classes through some form of sampling, e.g., over-sampling the minority or under-sampling the majority class. Under-sampling by discarding data can lead to loss of important information the model has to learn from. Over-sampling will cause an increased training time due to the increased size of the training set, and has also been shown to cause overfitting (Johnson and Khoshgoftaar 2019). In algorithm-level techniques, unlike data sampling methods, the learning or decision process is adjusted in a way that increases the importance of the minority class. Hybrid methods try to combine the other two data-level and algorithm-level approaches in various ways to handle the class imbalance problem (Seiffert et al. 2009)(Chen et al. 2021).

This paper can be categorized in algorithm-level approaches, and its main effort is to show that the hierarchical taxonomy of classes for an imbalanced dataset can lead classification learner to account for class imbalance. We highlight the positive effect of hierarchical taxonomies in the problem of imbalance flat classification which can be used as explicit regularization terms along with any other approach to deal with imbalanced data.

3 Proposed Methods

We propose two approaches to represent and integrate the hierarchical taxonomy as a prior knowledge into the loss function of a learning algorithm.

3.1 Symbolic-based Approach

To integrate the hierarchical taxonomy of the classes into the loss function, we first represent the taxonomy in symbolic logical constraints, and then inspired by (Xu et al. 2018) a differentiable semantic loss function is derived to capture how well the neural network is to satisfying the constraints on its output.

General Notation. We make use of concepts in propositional logic to formally define taxonomy and semantic loss. Boolean variables are written in uppercase letters (X, Y), and their instantiation ($X = 0$ or $X = 1$) are written in lowercase (x, y). We write sets of variables in bold uppercase ($\mathbf{X}, \mathbf{Y}$), and their joint instantiation in bold lowercase ($\mathbf{x}, \mathbf{y}$). A literal is a variable (X) or its negation ($\neg X$). A logical sentence (α or β) is created by variables and logical connectives ($\wedge$, $\vee$, etc.), and is also called a formula or constraint. A state $\mathbf{x}$ satisfies a sentence α, denoted $\mathbf{x} \models \alpha$, if the sentence evaluates to be true in that world, as defined in the usual way. The output vector of a neural network is noted by p. Each value in p represents a probability of an output in $[0, 1]$. The output vector of a set of sentences is noted by s. Each value in s represents a satisfaction value in $[0, 1]$.

Taxonomy. Each level of concept in a taxonomy is denoted $l_i, i \in [1, K]$, where K is node-based length of the taxonomy and l_1 indicates the leaves of the taxonomy, which is associated with the class labels. Each node in taxonomy except nodes in the leaves is denoted a_i. For example in Figure 1, in a taxonomy for multi-class classification, sentence α states that for a set of indicators $\mathbf{X} = \{X_1, .., X_6\}$, one and exactly one of X_3, X_4, X_5 must be true, with the rest being false. This statement indeed represents node a_2 of taxonomy in terms of its children/variables (X_3, X_4, X_5). To represent hierarchical nature of the taxonomy a set of variables $\mathbf{B} = \{B_1, B_2, .., B_{K-1}\}$ is defined over the taxonomy levels. A logical sentence β is created from variables $\mathbf{B}$ and logical connective $\wedge$.

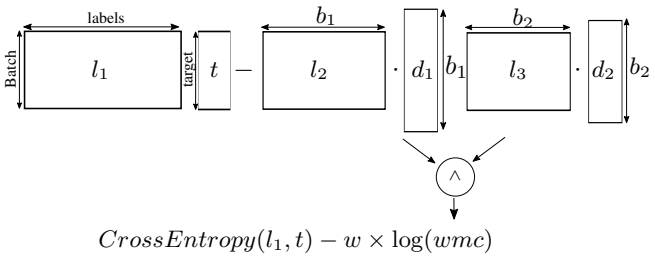


Figure 2: An illustration for supervised semantic loss

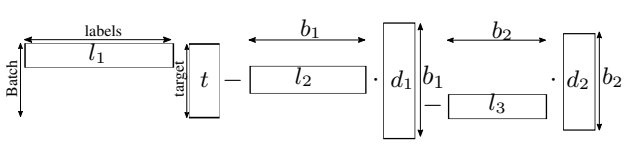
Semantic Loss. The semantic loss $L^s(\alpha, \beta, \mathsf{p}, \mathsf{s})$ is defined as a function of sentences (α,β) in propositional logic, defined over variables $\mathbf{X} = \{X_1, X_2, .., X_n\}$ and $\mathbf{B} = \{B_1, B_2, .., B_{K-1}\}$, a vector of probabilities p for variables $\mathbf{X}$, and a satisfaction vector s for variables $\mathbf{B} = \{B_1, B_2, .., B_{K-1}\}$. Element p_i denotes the predicted probability of variable X_i, and corresponds to a single output of the neural network. Element s_i denotes the satisfaction score of variable B_i, and corresponds to the output of a sentence α. Similar to (Xu et al. 2018), we define two examples of integrating semantic loss L^s into an existing loss function, simply as another regularization term, in supervised and semi-supervised fashions. More specifically, with some weight w, the new loss becomes

$$existing\_loss + w \cdot semantic\_loss.$$

Supervised-based Definition. For the Supervised-based definition, we assume all the training dataset are labeled, and hierarchical taxonomy is complete, i.e., for labeled class all the upper parents are given. Formally, for a class label cl_i, its $K - 1$ upper concepts in the taxonomy are given. With this assumption, let p be a vector of probabilities, one for each variable in $\mathbf{X}$, let α be a sentence over $\mathbf{X}$, and β be a sentence over $\mathbf{B}$.

$$L^s(\alpha, \beta, \mathsf{p}, \mathsf{s}) \propto -\log \prod_{\mathbf{y} \models \beta} \sum_{\mathbf{x} \models \alpha} \prod_{i:\mathbf{x} \models X_i} \mathsf{p}_i \prod_{i:\mathbf{x} \models \neg X_i} (1 - \mathsf{p}_i) \quad (1)$$

An illustration of Equation 1 is show in Figure 2, including target labels, training batch, and SDDs. Where target labels t are used in an existing data-driven loss function (e.g., cross entropy).



$$CrossEntropy(l_1, t) - w_1 \times \log(l_2 \cdot wmc(d_1)) - w_2 \times \log(l_3 \cdot wmc(d_2))$$

Figure 3: An illustration for semi-supervised semantic loss

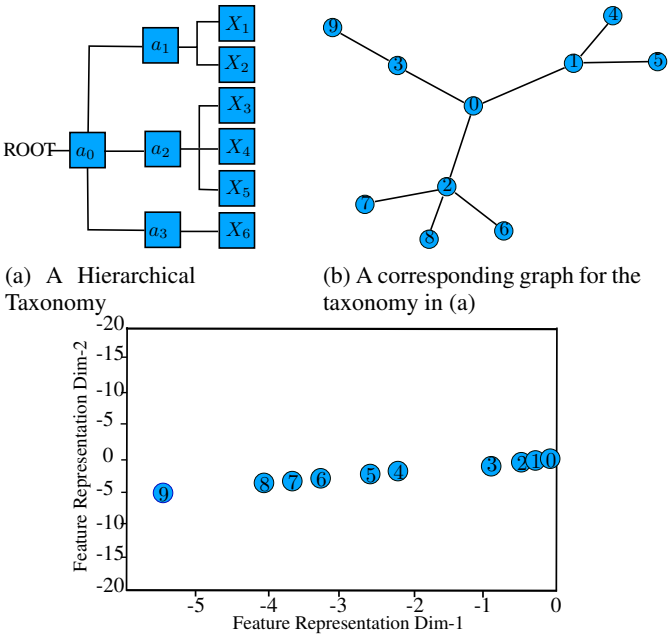
Semi-supervised-based Definition. There is a growing interest into utilizing unlabeled data to augment the predictive power of classifiers. In this section, we show integrating of hierarchical taxonomy with unlabeled data. In Semi-supervised-based definition, the assumption is that, there is unlabeled data and hierarchical taxonomy is not complete. The semantic loss is defined for unlabeled data using a incomplete taxonomy. The labeled data directly is used in an existing loss function (e.g., cross entropy). For the unlabeled data, we use the available deepest concepts/nodes from root, the upper node is considered in case of missing a lower node. In this definition of semantic loss, since there is no conflicts between different levels of concepts in the hierarchical taxonomy, there is no need for a sentence β over **B**. The intuition behind this is first; emphasizing information carried by unlabeled data, and then providing a level-based weighing for the incomplete taxonomy.

$$L^s(\alpha, \beta, \mathsf{p}, \mathsf{s}) \propto -\log \sum_{j \in \{1,k\}} \sum_{\mathbf{x} \models \alpha} \prod_{i:\mathbf{x} \models X_i} \mathsf{p}_i \prod_{i:\mathbf{x} \models \neg X_i} (1 - \mathsf{p}_i) \tag{2}$$

An illustration of Equation 2 is show in Figure 3, including training batch and SDDs.

Our goal is to develop a tractable loss for computing both semantic loss and its gradient. From propositional logic theories we know Model is a solution to a given propositional formula Δ, and Model Counting or #SAT is the problem of computing the number of models for Δ. In case of mapping literals of the variables to non-negative real-valued weights, we will have Weighted Model Counting (WMC) (Chavira and Darwiche 2008; Sang, Beame, and Kautz 2005). Indeed, the well-known task of model counting corresponds to the special case where all literal weights are 1 (and counts thus restricted to the natural numbers), whereas probabilistic inference (Prob) in a setting where all variables are independently assigned truth values at random restricts the weight function ω of WMC to values from [0, 1] such that weights of positive and negative literals for each var sum to one, i.e., for every variable υ, $\omega(\upsilon) \in [0, 1]$ and $\omega(\neg\upsilon) = 1 - \omega(\upsilon)$ (Kimmig, Van den Broeck, and De Raedt 2017).

From (Darwiche 2003) we know of differential circuit languages that compute WMCs, which are amenable to backpropagation. Following (Xu et al. 2018) we use the circuit compilation techniques in (Darwiche 2011),i.e., the Sentential Decision Diagram (SDD), to build a Boolean circuit representing semantic loss. Regarding two main properties of this circuit form SDD; determinism and decomposability, we can use it to compute both the values and the gradients of semantic loss in time linear in the size of the circuit (Darwiche and Marquis 2002).



(a) A Hierarchical Taxonomy

(b) A corresponding graph for the taxonomy in (a)

(c) 2-Dim plot of GCN representations for the nodes

Figure 4: An illustration for GCN encoding of the nodes in a taxonomy with labeled nodes $a_0, .., a_3$, and unlabeled nodes $X_1, .., X_6$.

3.2 GCN-based Approach

Graph Convolutional Networks is a powerful method presented for semi-supervised learning on graph-structured data (Kipf and Welling 2016), in which the authors consider the problem of classifying nodes (such as documents) in a graph (such as a citation network), where labels are only available for a small subset of nodes. Similarly, we define the problem of identifying representation for some of nodes in a graph, given the labels of other nodes. We consider a hierarchical taxonomy as a labeled graph and looking for GCN encoding of any external connected node to this graph. Figure 4 shows a simple illustration for the nodes of a taxonomy in a 2-Dim encoding.

one issue with GCN is memory required for encoding a big graph-structured data to provide representations for each node. Using GCN on whole the graph data avoids an explicit regularization with another supervised loss function (e.g., Cross Entropy). In this section we propose a method to inject hierarchical taxonomy of a classification task as a prior knowledge to the loss function through a batch-based Graph Convolutional Networks. A representation for a graph A in GCN is:

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{3}$$

where $\tilde{A} = A + L_N$ is the adjacency matrix of the undirected graph A with added self-connections. I_N is the identity matrix, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ and $W^{(l)}$ is a layer-specific trainable weight matrix. $\sigma(.)$ Denotes an activation function (we used

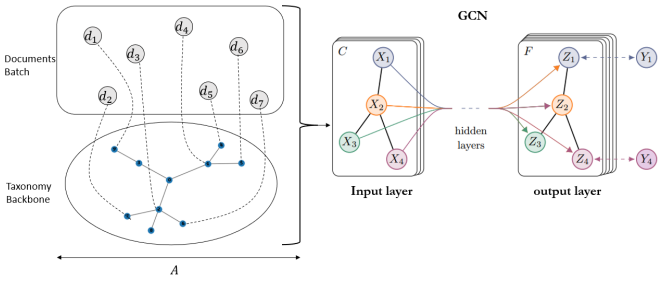


Figure 5: GCN

$ReLU$ in our experiments). $H^{(l)} \in \mathbb{R}$ is the matrix of activations in the l^{th} layer; $H^0 = X$, $H^2 = softmax(H^1)$.

We provide a taxonomy backbone graph for each batch which is consistent for all the batches and is generated from taxonomy tree. In this method we want the training algorithm (DNN) as well as relying on a supervised loss function, also consider a prior domain knowledge encoded in a taxonomy tree. Therefore, a batch includes a few documents from training data plus the taxonomy tree of the hierarchical categories of the classification which is used as a Backbone of a bigger graph A. Indeed, A is a graph generated by connecting documents of a batch to the taxonomy tree. The whole workflow of the end-to-end training is shown in Figure 5. The regularization term $\mathcal{L}_{reg}$ which is added explicitly into the existing loss function is defined as:

$$\mathcal{L}_{reg} = \| P - H \|_2^2 \tag{4}$$

The generated graph A is used to provide representations H for batch of document in the same space of the predicted probabilities from a supervised DNN. Euclidean distance is measured as the regularization loss to be added to the training loss. The final loss function is:

$$\mathcal{L} = \mathcal{L}_0 + w \times \mathcal{L}_{reg} \tag{5}$$

where $\mathcal{L}_0$ is a Cross Entropy loss and $\mathcal{L}_{reg}$ is the regularization loss.

4 Experimental Results

4.1 Data

We use two datasets of a private company with hight number of categories. A user query logs from Shopping Mall, including 84 classes, and a taxonomy in three levels from root to the leaves, i.e., level 1 with 18 domains, level 2 includes 45 intents, and level 3 with 84 sub-intent. We split data into training and testing datasets including 13962 and 1530 instances, respectively. Anther dataset is user query logs from Call Center Service with 134 classes. It has a hierarchical taxonomy with three levels; level 1 includes 5 domains, level 2 includes 24 intents, and level 3 (leaves/labels) includes 134 sub-intents. We split the dataset into training and testing data including 19214 and 1619 instances, respectively.

Method	Accuracy%	Macro Avg F1%	Weighted Avg F1%
Baseline	75	57	76
Symbolic-based	**76**	**61**	**77**
GCN-based	**76**	59	**77**

Table 1: A comparison of methods in Supervised fashion on Call Center Service dataset.

Method	Accuracy%	Macro Avg F1%	Weighted Avg F1%
Baseline	94	82	94
Symbolic-based	94	**84**	94
GCN-based	-	-	-

Table 2: A comparison of methods in Supervised fashion on Shopping Mall dataset.

4.2 Evaluation Measure

We use three measures *Accuracy*, *Macro Average F1-score*, and *Weighted Average F1-score* to evaluate obtained results. Specifically, for evaluating the effect of hierarchical taxonomy in imbalanced classification, we use *Macro Average F1-score*, which is an arithmetic mean of F1-scores per-class. It is not using weights (i.e., number of true labels of each class) for aggregation of F1-scores per-class, and this results in a bigger penalization when a model does not perform well with the minority classes, i.e., exactly what happens in imbalance classification.

TOBEDONE

5 Conclusion

TOBEDONE

Acknowledgments

TOBEDONE

References

Chavira, M., and Darwiche, A. 2008. On probabilistic inference by weighted model counting. *Artificial Intelligence* 172(6-7):772–799.

Chen, Z.; Duan, J.; Kang, L.; and Qiu, G. 2021. A hybrid data-level ensemble to enable learning from highly imbalanced dataset. *Information Sciences* 554:157–176.

Darwiche, A., and Marquis, P. 2002. A knowledge compilation map. *Journal of Artificial Intelligence Research* 17:229–264.

Portion	Method	Accuracy%	Macro Avg F1%	Weighted Avg F1%
20%	Baseline	68.99	46.56	69.05
	Symbolic-based	-	-	-
	GCN-based	**70.47**	**47.34**	**70.61**
30%	Baseline	-	-	-
	Symbolic-based	-	-	-
	GCN-based	-	-	-

Table 3: A comparison of methods in Semi-Supervised fashion on Call Center Service dataset.

Portion	Method	Accuracy%	Macro Avg F1%	Weighted Avg F1%
20%	Baseline	85.42	44.78	84.66
	Symbolic-based	**91.11**	**60.43**	**90.07**
	GCN-based	-	-	-
30%	Baseline	91.70	67.66	91.05
	Symbolic-based	**92.55**	**73.30**	**91.99**
	GCN-based	-	-	-
40%	Baseline	92.42	75.11	91.92
	Symbolic-based	**93.07**	**75.64**	**92.59**
	GCN-based	-	-	-

Table 4: A comparison of methods in Semi-Supervised fashion on Shopping Mall dataset.

Darwiche, A. 2003. A differential approach to inference in bayesian networks. *Journal of the ACM (JACM)* 50(3):280–305.

Darwiche, A. 2011. Sdd: A new canonical representation of propositional knowledge bases. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

Hasanin, T.; Khoshgoftaar, T. M.; Leevy, J. L.; and Bauder, R. A. 2019. Severely imbalanced big data challenges: investigating data sampling approaches. *Journal of Big Data* 6(1):1–25.

Johnson, J. M., and Khoshgoftaar, T. M. 2019. Survey on deep learning with class imbalance. *Journal of Big Data* 6(1):1–54.

Kimmig, A.; Van den Broeck, G.; and De Raedt, L. 2017. Algebraic model counting. *Journal of Applied Logic* 22:46–62.

Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Sang, T.; Beame, P.; and Kautz, H. A. 2005. Performing bayesian inference by weighted model counting. In *AAAI*, volume 5, 475–481.

Seiffert, C.; Khoshgoftaar, T. M.; Van Hulse, J.; and Napolitano, A. 2009. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 40(1):185–197.

Xu, J.; Zhang, Z.; Friedman, T.; Liang, Y.; and Broeck, G. 2018. A semantic loss function for deep learning with symbolic knowledge. In *International conference on machine learning*, 5502–5511. PMLR.