A APPENDIX

A.1 MULTI-LABEL CLASSIFICATION

In multi-label classification, every data instance can be assigned to multiple classes. It is relevant in many application domains. Many frameworks investigate the correlation/inter-dependency information among these multiple classes/ labels, including the weighted approximate pairwise ranking loss (WARP) Weston et al. (2011), calibrated separation ranking loss (CSRL) Guo & Schuurmans (2011) and the max-margin learning methods with pairwise ranking loss Hariharan et al. (2012). Due to noise or crowd-sourcing, many multi-label applications encounter incomplete labels frequently, where only a subset of true labels on some training instances is available. The performance of multi-label learning methods with missing labels largely depends on learned label correlations in the training phase Bucak et al. (2011); Yu et al. (2014); Yang et al. (2016). The major assumption in these methods is the availability of all the labels on a subset of training data (at least). Hence, tackling the zero-shot learning setting where some labels are completely missing from the training instances is more challenging and cannot be addressed with these methods.

A.2 ZERO-SHOT LEARNING RELATED WORKS

Existing ZSL methods generally follow two frameworks based on the relationship between the (seen and unseen) classes.

In embedding-space based methods like Farhadi et al. (2009); Socher et al. (2013); Lampert et al. (2009), each class/label is represented by an attribute vector representing the shared attributes between classes Farhadi et al. (2009) or a word semantic embedding Lampert et al. (2009) learned from a large text corpus. This approach embeds both seen and unseen classes into a shared vector space. By utilizing the fully labeled samples from seen classes, a sample embedding function can be learned to project a sample from its feature space into the shared embedding space. Since the classes (seen and unseen) are related in the shared space, the function learned from the seen classes can also work for the unseen classes.

In the second type of the methods, class-similarity-based approaches, the availability of the similarity among all seen and unseen classes is explored Lampert et al. (2014); Norouzi et al. (2013). In this approach, a categorical classifier for the seen classes is learned from the labeled data at first. Then, the learned classifier can calculate its probability distribution on the seen classes for a test sample. Finally, the probability distribution of the unseen classes is computed based on the class similarity. These two methods adopt a two-step strategy, and an intermediate transformation in the test stage is required Guo et al. (2017) resulting in high time complexity and low performance due to the loss of the information. Many algorithms and methods investigate how to bridge the gap between seen and unseen categories resulting in a good performance in the testing phase. These methods all rely on similarities between seen and unseen classes and ignore the benefit of considering the *relationship/structure* among classes. For instance, the low value of co-occurrence features for seen classes like fish and giraffe can occur among unseen classes like whale and zebra. Therefore, when the classifier recognizes the existence of a zebra in a test image, it can assign a low probability of existing a whale in that image considering similar feature representations of fish and whale families and their relationship. This type of information can be very beneficial in recognizing unseen classes. We build up our approach based on this relationship factor and present it by investigating ZSL in multi-label classification.

Many of the prior works on zero-shot learning model multi-class learning problems Socher et al. (2013); Norouzi et al. (2013); Changpinyo et al. (2016). Some of these works investigate different strategies for transferring embedding Higgins et al. (2005), and some of them explore differently information sources Mensink et al. (2014); Akata et al. (2015b). There are some existing works Zhang et al. (2016); Mensink et al. (2014); Sandouk & Chen (2016); Fu et al. (2015) that directly apply the zero-shot multi-class problem to the zero-shot multi-label problem. However, these frameworks ignore the structure of the output (label matrix), which is extremely important as proved by the better performance of multi-label algorithms Chen & Lin (2012); Yu et al. (2014); Kapoor et al. (2012); Rai et al. (2015); Bhatia et al. (2015). In applications like recommender systems, inductive matrix completion methods are employed. The strong assumption in these methods is that the features for both training examples and labels are provided. However, providing a pre-defined set of good features may be impossible. Our method overcomes this need by using co-occurrence word-embedding Mikolov et al. (2013) features like word2vec features.

In Fu et al. (2015), the multi-label zero-shot learning is addressed by mapping images into the semantic word space, and all possible combinations of the outputs are considered in the testing phase. This framework's time and space complexity does not make it a good candidate for large data sets. Mensink et al. (2014) demonstrate unseen class classifiers as weighted sums of seen class classifiers. Hence, the weights are approximated from different types of co-occurrence statistics. This method treats the unseen class classifiers separately and lacks in considering the correlations/dependencies among the classes. Recently, Gaure et al. (2017) proposed a generative probabilistic framework to leverage the co-occurrence statistics of the seen labels for multi-label zero-shot learning. This method highly depends on the auxiliary resources providing quality label co-occurrence statistics. Lee et al. (2017) modeled the label relations by defining a knowledge graph based on WordNet hierarchy and then transferred the confidence scores from the seen to unseen labels through the graph. The performance of this method also largely relies on the quality of the knowledge graph. Ye & Guo (2018) tried to decrease this dependency on auxiliary resources. In this approach, existing label embeddings are projected into a low-dimensional semantic space to automatically retrieve better label relations for knowledge transfer between seen and unseen classes. It can still exploit auxiliary information for additional help.

A.3 ALGORITHM

We present the double oracle method in this section. The algorithm requires our feature representations (joint, layer-wise, taxonomy, and pairwise features) and the learning parameters θ as input.

Algorithm 1 Double Oracle Algorithm for Min-Max ML-ZSL Equilibria (training phase)

```
Require: Unary features $\{\phi^i(\cdot, \mathbf{x})\}$; Pairwise features $\{\phi^{i,j}(\cdot, \cdot, \mathbf{x})\}$;
    Parameters $\boldsymbol{\theta}$; Initial label $\mathbf{y}_{\text{initial}}$
Ensure: Nash equilibrium, $(P_{mini}, P_{maxi})$
$\mathcal{S}_{mini} \leftarrow \mathcal{S}_{maxi} \leftarrow \{\mathbf{y}_{\text{initial}}\}$
repeat
    $(P_{mini}, P_{maxi}, V_{maxi}) \leftarrow \text{solveGame}(\boldsymbol{\Psi}(\mathcal{S}_{maxi}), \text{loss}_{\text{Ham}}(\mathcal{S}_{maxi}, \mathcal{S}_{mini}))$
    $(\mathbf{y}_{maxi\text{-}new}, \mathbf{V}_{\text{max}}) \leftarrow \max_{\mathbf{y_{maxi}}} \mathbb{E}_{\mathbf{Y}_{mini} \sim \mathbf{P}_{mini}}[\text{loss}_{\text{Ham}}(\mathbf{y}_{maxi}, \mathbf{y}_{mini}) + \boldsymbol{\Psi}(\mathbf{y_{maxi}})]$
    if $(V_{maxi} \neq V_{\text{maximum}})$ then
        $\mathcal{S}_{maxi} \leftarrow \mathcal{S}_{maxi} \cup \mathbf{y}_{maxi\text{-}new}$
    end if
    $(P_{mini}, P_{maxi}, V_{mini}) \leftarrow \text{solveGame}(\boldsymbol{\Psi}(\mathcal{S}_{maxi}), \text{loss}_{\text{Ham}}(\mathcal{S}_{maxi}, \mathcal{S}_{mini}))$
    $(\mathbf{y}_{mini\text{-}new}, \mathbf{V}_{\text{min}}) \leftarrow \min_{\mathbf{y}_{mini}} \mathbb{E}_{\mathbf{y}_{maxi} \sim \mathbf{P}_{maxi}}[\text{loss}_{\text{Ham}}(\mathbf{y}_{maxi}, \mathbf{y}_{mini})]$
    if $(V_{mini} \neq V_{\text{minimum}})$ then
        $\mathcal{S}_{mini} \leftarrow \mathcal{S}_{mini} \cup \mathbf{y}_{mini\text{-}new}$
    end if
until $V_{maxi} = V_{\text{maximum}} = V_{mini} = V_{\text{minimum}}$
Return$(P_{mini}, P_{maxi})$
```

The set of strategies (label vector assignments) S for each player is initialized randomly. The game matrix is defined by leveraging the initial set of strategies for maximizer S_{maxi} and minimizer S_{mini}. The termination condition $v_{maxi} = v_{maximum} = v_{mini} = v_{minimum}$ presents the saddle point in the algorithm where the maximizer/minimizer cannot improve $v_{maxi/mini}$ anymore, it is called $v_{maximum/minimum}$. The equilibria reach the optimal solution at this point. As it is presented in algorithm 1, the general structure of the algorithm relies on two main procedures: *solveGame* and finding the best possible strategy/ response (i.e., argmax or argmin). This minimax zero-sum game can be solved in polynomial time using linear programming Neumann et al. (1944). It is illustrated as *solveGame* in the algorithm. $\boldsymbol{\Psi}(\mathcal{S}_{maxi})$ presents the potentials for the label vectors of as $\boldsymbol{\Psi}(\mathcal{S}_{maxi}) \triangleq \{\Psi(\mathbf{y_{maxi}}, \mathbf{x})\}_{\mathbf{y_{maxi}} \in \mathcal{S}_{maxi}}$.

A.4 EQ. 8 TO EQ. 9 TRANSFORMATION PROOF

In this section, we provide the detailed step-by-step transformation from equation 8 to equation 9 : The transformation steps a-c are described in the following: a) Flipping the min and max order using the minimax duality. We show the constraints (eq. 8 and 9) as Ξ for the sake of space.

$$\min_{P_{mini}(\mathbf{Y}_{mini}|x)} \max_{P_{maxi}(\mathbf{Y}_{maxi}|x)} \mathbb{E}_{x\sim P_{Data};\mathbf{Y}_{mini}|x\sim P_{mini};\mathbf{Y}_{maxi}|x\sim P_{maxi}} [\text{loss}(\mathbf{Y}_{mini}, \mathbf{Y}_{maxi})] \quad \text{s.t.} \tag{13}$$

$$\mathbb{E}_{x\sim P_{Data};\mathbf{Y}_{maxi}|x\sim P_{maxi}} \left[\sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i}_{maxi})\right] = \mathbb{E}_{(x,\mathbf{Y})\sim P_{Data}} \left[\sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i})\right] \tag{14}$$

$$\mathbb{E}_{x\sim P_{Data};\mathbf{Y}_{maxi}|x\sim P_{maxi}} \left[\sum_{i\neq j} \phi^{i,j}(x, \mathbf{Y}^{i}_{maxi}, Y^{j}_{maxi})\right] \leq \mathbb{E}_{(x,\mathbf{Y})\sim P_{Data}} \left[\sum_{i\neq j} \phi^{i,j}(x, \mathbf{Y}^{i}, Y^{j})\right] \tag{15}$$

$$\overset{(a)}{=} \max_{P_{maxi}(\mathbf{Y}_{maxi}|x)} \min_{P_{mini}(\mathbf{Y}_{mini}|x)} \mathbb{E}_{x\sim P_{Data};\mathbf{Y}_{mini}|x\sim P_{mini};\mathbf{Y}_{maxi}|x\sim P_{maxi}} [\text{loss}(\mathbf{Y}_{mini}, \mathbf{Y}_{maxi})] \quad \text{s.t. } \Xi \tag{16}$$

$$\begin{aligned}\overset{(b)}{=} & \max_{P_{maxi}(\mathbf{Y}_{maxi}|x)} \min_{\theta^{i},\theta^{(i,j)}} \min_{P_{mini}(\mathbf{Y}_{mini}|x)} \mathbb{E}_{(x,\mathbf{Y})\sim \tilde{P};\mathbf{Y}_{mini}|x\sim P_{mini};\mathbf{Y}_{maxi}|x\sim P_{maxi}} \Bigg[\text{loss}(\mathbf{Y}_{mini}, \mathbf{Y}_{maxi}) + \\ & \theta^{i\,\mathrm{T}} \left(\sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i}_{maxi}) - \sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i})\right) + \theta^{(i,j)\,\mathrm{T}} \left(\sum_{i\neq j} \phi^{(i,j)}(x, \mathbf{Y}^{i}_{maxi}, Y^{j}_{maxi}) - \sum_{i\neq j} \phi^{(i,j)}(x, \mathbf{Y}^{i}, Y^{j})\right)\Bigg]\end{aligned} \tag{17}$$

$$\begin{aligned}\overset{(c)}{=} & \min_{\theta^{i},\theta^{(i,j)}} \max_{P_{maxi}(\mathbf{Y}_{maxi}|x)} \min_{P_{mini}(\mathbf{Y}_{mini}|x)} \mathbb{E}_{(x,\mathbf{Y})\sim P_{Data};\mathbf{Y}_{mini}|x\sim P_{mini};\mathbf{Y}_{maxi}|x\sim P_{maxi}} \Bigg[\text{loss}(\mathbf{Y}_{mini}, \mathbf{Y}_{maxi}) + \\ & \theta^{i\,\mathrm{T}} \left(\sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i}_{maxi}) - \sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i})\right) + \theta^{i,j\,\mathrm{T}} \left(\sum_{i\neq j} \phi^{(i,j)}(x, \mathbf{Y}^{i}_{maxi}, Y^{j}_{maxi}) - \sum_{i\neq j} \phi^{i,j}(x, \mathbf{Y}^{i}, Y^{j})\right)\Bigg]\end{aligned} \tag{18}$$

$$\begin{aligned}\overset{(d)}{=} & \min_{\theta^{i},\theta^{(i,j)}} \mathbb{E}_{(x,\mathbf{Y})\sim P_{Data}} \max_{P_{maxi}(\mathbf{Y}_{maxi}|x)} \min_{\hat{P}(\mathbf{Y}_{mini}|x)} \mathbb{E}_{\mathbf{Y}_{mini}|x\sim P_{mini};\mathbf{Y}_{maxi}|x\sim P_{maxi}} \Bigg[\text{loss}(\mathbf{Y}_{mini}, \mathbf{Y}_{maxi}) + \\ & \theta^{i\,\mathrm{T}} \left(\sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i}_{maxi}) - \sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i})\right) + \theta^{i,j\,\mathrm{T}} \left(\sum_{i\neq j} \phi^{(i,j)}(x, \mathbf{Y}^{i}_{maxi}, Y^{j}_{maxi}) - \sum_{i\neq j} \phi^{i,j}(x, \mathbf{Y}^{i}, Y^{j})\right)\Bigg]\end{aligned} \tag{19}$$

$$\begin{aligned}\overset{(e)}{=} & \min_{\theta^{i},\theta^{(i,j)}} \mathbb{E}_{(x,\mathbf{Y})\sim \tilde{P}} \min_{P_{mini}(\mathbf{Y}_{mini}|x)} \max_{P_{maxi}(\mathbf{Y}_{maxi}|x)} \mathbb{E}_{\hat{\pi}|x\sim P_{mini};\mathbf{Y}_{maxi}|x\sim P_{maxi}} \Bigg[\text{loss}(\mathbf{Y}_{mini}, \mathbf{Y}_{maxi}) + \\ & \theta^{i\,\mathrm{T}} \left(\sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i}_{maxi}) - \sum_{i=1}^{n} \phi^{i}(x, \mathbf{Y}^{i})\right) + \theta^{(i,j)\,\mathrm{T}} \left(\sum_{i\neq j} \phi^{(i,j)}(x, \mathbf{Y}^{i}_{maxi}, Y^{j}_{maxi}) - \sum_{i\neq j} \phi^{(i,j)}(x, \mathbf{Y}^{i}, Y^{j})\right)\Bigg].\end{aligned} \tag{20}$$

b) Introducing the Lagrange dual variable $\theta = [\theta^{i}, \theta^{i,j}]$ for the constraints on unary (nodes) and pairwise (edges) features.

c) The domain of $P_{maxi}(\mathbf{Y}_{maxi}|x)$ is a compact convex set, where the domain of θ is convex (i.e. $\mathbb{R}^{d}$ where d is the number of features). The objective is concave on $P_{maxi}(\mathbf{Y}_{maxi}|x)$ since a non-negative linear combination of minimums of affine function is concave, while it is convex on θ. Based on Sion's minimax theorem Sion (1958), a strong duality holds and we can flip the order of $P_{maxi}(\mathbf{Y}_{maxi}|x)$ and θ in the optimization.

d) Pushing the expectation over the empirical distribution outside the inner max-min.

e) Applying the minimax duality Von Neumann & Morgenstern (1945) again to flip the optimization order of the inner min-max, resulting in Eq.9 in the paper.

Our method gets the benefit of probabilistic min-cut output for knowledge transfer. The node and edges weights and features direct the min-cut in the right direction. When there is strong/weak similarity between unseen and seen classes (nodes), the potentials of the edges and nodes ($\theta * \phi$) get bigger/smaller, and the min-cut algorithm puts them in connection with node 1/0.

A.5 MORE EXPERIMENTS

We run another set of experiments considering Costa Guo et al. (2017), Conse Norouzi et al. (2013), Fast0tag Zhang & Saligrama (2015) Softmax, TAEP Ye & Guo (2019), and TAEP-C Ye et al. (2015)

methods to challenge the efficiency of our approach in comparison with more methods on more data-sets. We report data-set information and the experimental results in tables 4 and 5. For the experiment setting, we follow Zhang & Saligrama (2015) for all the experiments. Table 5 presents the experimental results of running seven ZSL methods on four well-known datasets. As it is presented in this table, Min-Max outperforms Costa, Fast0Tag, TAEP, TAEP-c, softmax, and Conse methods by 0.07, 0.16, 0.13, 0.09, 0.21, 0.11 in average, respectively. Hence, our approach outperforms other methods at least by 0.07, which is promising considering the size of the data sets. We run a set of experiments following Wang et al. (2018) experiment setting on multi-class ZSL leveraging our multi-label ZSL. This set of experiments considers different neural network architectures (Inception-V1, AlexNet, and ResNet-50) and reports the top-k accuracy. We report data-set information and the experimental results in tables 6 and 7. These tables show that the Min-Max method outperforms other methods considering different deep-neural network architecture and datasets on both ZSL and Generalized ZSL. We run extensive experiments and benchmark our proposed method, Min-Max, with structural SVM (SSVM) Taskar et al. (2005) implemented using SVM-Struct Vedaldi (2011). The feature representation in this set of experiments includes unary and pairwise features. In addition, we use features from Mulan dataset Tsoumakas et al. (2011) as sample features to present joint features using equation 1. The layer-wise features, as defined in equation 2 are computed For image data-sets. We leverage VGG 19 Convolutional neural network(CNN) Simonyan & Zisserman (2014) and extract the features from different convolution layers (1, 3, 6, 8, 11, 13, 15, 18, 20, 22) for every example. One novelty of our method is exploring which convolutional layers provide more meaningful features for ZSL and the best combination of these layers. The unary features are the concatenation of joint, layer-wise, and taxonomy features. Table 1 presents the number of seen and unseen labels for every experiment. It is considerable that the testing phase includes both seen and unseen classes while the loss is calculated only based on the prediction of unseen classes. The nodes and edges in the graph made by these seen and unseen classes carry more informative information resulting in more accurate graph cuts and generalizations. We run two experiments to highlight that our methods target non-image (experiment 1) and image domains (experiment 2), and our min-max ZSL outperforms other methods. Besides, it helps us to highlight the positive effect of adding hierarchical features for image datasets.:

Name	#Features	# Seen	# Unseen	#Instances
SUN	102	645	72	14340
CUB	312	150	50	11788
AwA	85	40	10	30475
aPY	64	20	12	18627

Table 4: The Statistics of the four zero-shot learning datasets.

Experiments applying unary (joint+layer-wise) and pairwise features: We run experiments considering three methods, Costa Mensink et al. (2014), SSVM and Min-Max. There is no taxonomy information or layer-wise features for non-image datasets. Therefore, we ignore them in our feature representation. We report data-set characteristics and the experimental results in table 8.

Experiments applying unary (joint+ CNN layer-wise+taxonomy) and pairwise features: This experiment includes two sub-experiments to illustrate the role of taxonomy features and CNN layers. Both sub-experiments are applied to image datasets. We employ the taxonomy structure of VOC2006, presented in figure 4 as the taxonomy structure for nine classes: Bicycle, Motorbike, Bus, Car, Cow, Sheep, Cat, Dog, and Horse. We extract images from VOC2006 and NUS-WIDE datasets, which have at least two instances from these classes. The seen and unseen classes are presented in table 9 for every experiment. The training is done on seen classes, and the testing is applied to samples from

Name	SUN	AwA	CUB	aPY	Average
Costa	0.35	0.37	0.35	0.34	0.35
Fast0Tag	0.34	0.32	0.33	0.46	0.36
TAEP	0.39	0.35	0.34	0.43	0.38
TAEP-C	0.36	0.33	0.37	0.39	0.35
Softmax	0.27	0.30	0.31	0.37	0.33
ConSE	0.35	0.39	0.36	0.41	0.37
BiAm	0.38	0.39	0.40	0.43	0.4
Min-Max	0.44	0. 42	0.46	0.48	0.44

Table 5: The F1-score over 10 runs on four datasets.

			Hit@k (%)				
Test Set	Model	ConvNets	1	2	5	10	20
2-hops	ConSE	Inception-v1	8.3	12.9	21.8	30.9	41.7
	SYNC	Inception-v1	10.5	17.7	28.6	40.1	52.0
	EXEM	Inception-v1	12.5	19.5	32.3	43.7	55.2
	SEKG	Inception-v1	18.5	31.3	50.1	62.4	72.0
	SEKG	ResNet-50	19.8	33.3	**53.2**	65.4	74.6
	Min-Max	Inception-v1	19.7	33.5	51.0	62.7	72.6
	Min-Max	ResNet-50	**20.3**	**34.1**	**54.9**	**65.9**	**75.4**
3-hops	ConSE	Inception-v1	2.6	4.1	7.3	11.1	16.4
	SYNC	Inception-v1	2.9	4.9	9.2	14.2	20.9
	EXEM	Inception-v1	3.6	5.9	10.7	16.1	23.1
	SEKG	Inception-v1	3.8	6.9	13.1	18.8	26.0
	SEKG	ResNet-50	4.1	7.5	14.2	20.2	27.7
	Min-Max	Inception-v1	4.3	7.1	14.2	19.5	26.8
	Min-Max	ResNet-50	**4.9**	**8.6**	**15.5**	**21.5**	**29.1**
All	ConSE	Inception-v1	1.3	2.1	3.8	5.8	8.7
	SYNC	Inception-v1	1.4	2.4	4.5	7.1	10.9
	EXEM	Inception-v1	**1.8**	2.9	5.3	8.2	12.2
	SEKG	Inception-v1	1.7	3.0	5.8	8.4	11.8
	SEKG	ResNet-50	**1.8**	**3.3**	**6.3**	**9.1**	**12.7**
	Min-Max	Inception-v1	2.1	4.2	6.7	10.1	13.4
	Min-Max	ResNet-50	**2.3**	**3.9**	**7.4**	**10.3**	**13.5**

Table 6: Results on ImageNet. Top-k accuracy for different models when testing on only unseen classes.

			Hit@k (%)				
Test Set	Model	ConvNets	1	2	5	10	20
2-hops (+1K)	DeViSE	AlexNet	0.8	2.7	7.9	14.2	22.7
	ConSE	AlexNet	0.3	6.2	17.0	24.9	33.5
	SEKG	Inception-v1	7.9	18.6	39.4	53.8	65.3
	SEKG	ResNet-50	9.7	20.4	42.6	57.0	68.2
	Min-Max	Inception-v1	8.7	19.7	41.0	54.9	66.9
	Min-Max	ResNet-50	**11.2**	**23.2**	**43.8**	**57.9**	**69.7**
3-hops (+1K)	DeViSE	AlexNet	0.5	1.4	3.4	5.9	9.7
	ConSE	AlexNet	0.2	2.2	5.9	9.7	14.3
	SEKG	Inception-v1	1.9	4.6	10.9	16.7	24.0
	SEKG	ResNet-50	2.2	5.1	11.9	18.0	25.6
	Min-Max	Inception-v1	2.3	4.9	11.8	18.5	25.8
	Min-Max	ResNet-50	**3.3**	**6.5**	**12.8**	**19.5**	**26.4**
All (+1K)	DeViSE	AlexNet	0.3	0.8	1.9	3.2	5.3
	ConSE	AlexNet	0.2	1.2	3.0	5.0	7.5
	SEKG	Inception-v1	0.9	2.0	4.8	7.5	10.8
	SEKG	ResNet-50	1.0	2.3	5.3	8.1	11.7
	Min-Max	Inception-v1	2.1	3.8	6.5	9.1	12.3
	Min-Max	ResNet-50	**2.6**	**4.1**	**7.1**	**9.8**	**13.2**

Table 7: Results on ImageNet.Top-k accuracy for different models, testing on both seen and unseen classes.

both seen and unseen classes. We report the results of this experiment in table 9. The numbers in the seen and unseen columns indicate the class numbers in the VOC2006 taxonomy presented in figure 4.

This experiment also considers extracting features from different CNN layers to investigate which layers provide more informative features for transformation. We only report the best performance using "CNN 6, 33" (using layers 6 and 33 of the MatConvNet) and a middle-range performance using "CNN 13, 21" to express the significance of layer selection.

Name	Domain	#Instances	#Features	# Labels	#Seen	# Unseen	Costa	SSVM	Min-Max	#cuts
Bibtex	text	7395	1836	159	95	64	-	0.33	0.39	12.2
Bookmarks	text	87856	2150	202	121	81	-	0.38	0.43	11.8
Birds	audio	645	260	19	11	8	-	0.34	0.42	10.2
CAL500	music	502	68	174	104	70	-	0.35	0.40	12.8
Emotions	music	593	72	6	3	3	-	0.42	0.49	13.6
Flags	images	194	19	7	4	3	-	0.41	0.50	8.6
Scene	images	2407	294	6	3	3	0.36	0.39	0.42	10.2
Yeast	biology	2417	103	14	8	6	-	0.41	0.47	11.9
NUS-WIDE	images	269648	128	81	61	20	0.39	0.43	0.51	12.6
						Average	-	0.38	0.45	11.5

Table 8: Multi-label dataset information and F-score results on the test set for SSVM, Costa, and our Min-Max approach.

Name	Seen	Unseen	SSVM ZSL	Min-Max ZSL	#cuts
	CNN 13 , 21	CNN 13 , 21			
VOC2006	{4,7,13,17,19}	{5,8,14,18}	0.301	0.353	10.2
NUS-WIDE	{4,7,13,17,19}	{5,8,14,18}	0.321	0.384	14.2
VOC2006	{5,8,14,18}	{4,7,13,17,19}	0.332	0.378	7.2
NUS-WIDE	{5,8,14,18}	{4,7,13,17,19}	0.312	0.353	19.1
VOC2006	{4,5,17,18}	{7,8,19}	0.394	0.349	21.8
VOC2006	{7,8,19}	{4,5,17,18}	0.373	0.381	14.5
		Average	0.339	0.383	14.5
	CNN 6 , 33	CNN 6 , 33			
VOC2006	{4,7,13,17,19}	{5,8,14,18}	0.391	0.423	12.1
NUS-WIDE	{4,7,13,17,19}	{5,8,14,18}	0.370	0.464	9.5
VOC2006	{5,8,14,18}	{4,7,13,17,19}	0.404	0.458	14.3
NUS-WIDE	{5,8,14,18}	{4,7,13,17,19}	0.382	0.443	22.3
VOC2006	{4,5,17,18}	{7,8,19}	0.436	0.459	19.4
VOC2006	{7,8,19}	{4,5,17,18}	0.425	0.461	13.2
		Average	0.393	0.451	17.1

Table 9: The F-score for SSVM ML-ZSL and our Min-Max ML-ZSL approach.

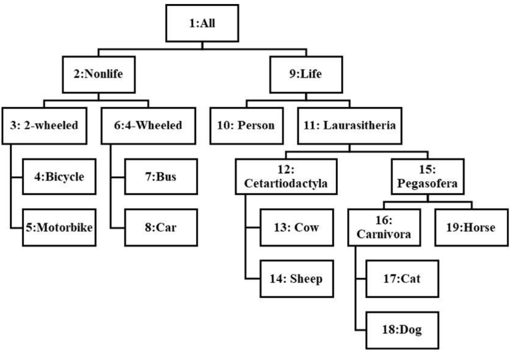

Figure 4: The VOC2006 taxonomy.

A.6 DISCUSSION

As shown in tables 8 and 9, Min-Max always provides better performance in comparison with other methods. Considering the experiment results in table 8, Min-Max provides a minimum of 0.02 and a maximum 0.12 loss less than SSVM over different datasets. The difference is also considerable in experiments in which taxonomy features are considered (table 9). In experiment 9, Min-Max outperform other methods by minimum 0.05 and maximum 0.09 using "CNN 6, 33". It beats other methods by a minimum of 0.07 and a maximum of 0.09 using "CNN 13, 21". This advantage results from the Min-Max framework's ability to incorporate correlations between variables into its prediction as a graph min-cut. Our experiments show that the lower layers of CNN do not provide any information in ZSL prediction. Combining top layers with other layers provides better performance than just using top layers. The best performance is achieved by applying features from CNN layers 6 and 33. This experiment demonstrates the benefit of using features from the bottom and top layers of CNN to extract information regarding the structure of the sample. The degrading performance of Softmax and SSVM using "CNN 13, 21" compared to "CNN 6, 33" is worth mentioning. The promising performance of our method relies on three main keys: a) Modeling the problem as a graph representation and transferring classes relationship in addition to the classes themselves. The class relationship in our feature representation relies on word2vec, layer-wise, and taxonomy feature representation. b) In the training phase, the classifier is trained considering the worst data sample (the sample which maximizes/minimizes the loss value). Hence, the Min-Max method provides robust predictions. c) Considering taxonomy features. This feature representation is more beneficial because of graph representation. The learning parameter θ is already trained and adjusted in the training phase. The potential value ($\theta * features$) strengthens the nodes and edge weights in their taxonomy path, and It would lead the min-cut to make a right cut. The same reasoning exists for pairwise feature representations (classes relationship). The time complexity of Min-Max is reported by considering the average number of required min-cuts for a double oracle algorithm to provide the final result. In the first experiment, it is less than 12 on average, as it is presented in table 8. Thus, we can conclude from table 8 that the inference from scratch is roughly 12 times slower than SSVM and other methods. For the second experiment, which leverages the taxonomy features, the number of cuts is roughly 7-11 min-cuts using CNN 6, 33. During training, however, the strategies from the previous equilibria can be cached and reused, making training time comparable to other methods in practice, and showing the algorithm's efficient time complexity.

Name	Seen	Unseen	SSVM	Softmax	Min-Max	#cuts
	ResNet 14 , 25	ResNet 14, 25				
VOC2006	{4,7,13,17,19}	{5,8,14,18}	0.398	0.376	0.437	8.9
NUS-WIDE	{4,7,13,17,19}	{5,8,14,18}	0.430	0.408	0.465	8.5
VOC2006	{5,8,14,18}	{4,7,13,17,19}	0.423	0.378	0.463	7.5
NUS-WIDE	{5,8,14,18}	{4,7,13,17,19}	0.399	0.371	0.417	9.0
VOC2006	{4,5,17,18}	{7,8,19}	0.394	0.407	0.425	9.8
VOC2006	{7,8,19}	{4,5,17,18}	0.426	0.397	0.453	9.1
		Average	0.412	0.390	0.443	8.8
	ResNet 7 , 33	ResNet 7 , 33				
VOC2006	{4,7,13,17,19}	{5,8,14,18}	0.466	0.441	0.487	11.3
NUS-WIDE	{4,7,13,17,19}	{5,8,14,18}	0.457	0.471	0.489	8.9
VOC2006	{5,8,14,18}	{4,7,13,17,19}	0.471	0.467	0.498	9.8
NUS-WIDE	{5,8,14,18}	{4,7,13,17,19}	0.436	0.423	0.486	7.1
VOC2006	{4,5,17,18}	{7,8,19}	0.423	0.439	0.465	11.2
VOC2006	{7,8,19}	{4,5,17,18}	0.441	0.432	0.474	8.7
		Average	0.449	0.467	0.483	9.5

Table 10: The F-score for SSVM, Softmax, and our Min-Max approach (with the average number of cuts), employing ResNet.

Experiments applying unary (joint+ ResNet layer-wise+taxonomy) and pairwise features: We repeat the experiment in table 10 using different layers of ResNet He et al. (2016). As it is presented in table 10, ResNet layers "7, 33" provides better performance in compare with ResNet layers "14, 25". Our approach, Min-Max, provides better performance with comparing with other methods. The experimental results prove our claim on the better feature representation and efficiency of lower and top CNN layers in ZSL.

A.7 MORE EXPERIMENT ON SHARPNESS VALUE

In this experiment, we compute the sharpness value of different baselines considering $\epsilon = 10^{-3}$ in the sharpness formula 11. As it is presented in table 11, the Min-Max method still presents a much less sharpness value compared with the other baselines on three datasets. As presented in table 11, the methods that provide better performance in the ZSL setting have less sensitivity (sharpness).

Method/Datasets	VOC2007	VOC 2012	NUS-1000
$\epsilon = 5.10^{-4}$			
ConSE	53.32 ± 5.64	65.52 ± 3.94	65.37 ± 3.86
LatEm	-	59.67 ± 4.53	60.94 ± 5.58
Fast0Tag	48.87 ± 4.43	55.34 ± 4.14	50.76 ± 3.57
DMP	45.12 ± 3.87	48.37 ± 5.61	58.63 ± 4.52
TAEP-C	42.21 ± 3.32	44.83 ± 4.38	53.42 ± 4.59
LESA	36.12 ± 4.23	35.64 ± 3.84	48.29 ± 5.49
GEN	32.68 ± 4.32	30.54 ± 3.84	42.51 ± 3.61
BiAM	29.74 ± 4.39	28.87 ± 4.11	38.86 ± 5.10
Min-Max(ours)	26.86 ± 4.12	27.65 ± 3.89	36.59 ± 5.62

Table 11: Sharpness value considering $\epsilon = 5.10^{-4}$.

A.8 DATASETS:

In addition to Mulan, VOC 2006, and NUS-WIDE datasets, we consider four more datasets.

Animals with Attributes (AwA) AWA Lampert et al. (2014) is a widely-used dataset for zero-shot learning. It contains 30,475 images from 50 different animal classes with at least 92 labeled examples per class. The dataset creators provide a standard split into 40 seen and 10 unseen classes.

Caltech-UCSD-Birds-200-2011 (CUB) CUB Tsai et al. (2017) includes a large number of classes and attributes, containing 11,788 images from 200 different types of birds annotated with 312 attributes. Akata et al. (2015a) introduced the first zero-shot split of CUB with 150 seen classes and 50 unseen classes.

SUN Attribute(SUN) SUN Patterson & Hays (2012) is a medium-scale in the number of images dataset containing 14,340 images from 717 types of scenes annotated with 102 attributes. We follow the standard split presented in Lampert et al. (2014), containing 645 seen classes (where 65 classes are used for validation) and 72 unseen classes.

Attribute Pascal and Yahoo (aPY) It is a small dataset with 64 attributes and 32 classes Farhadi et al. (2009). We adopt the split in Xian et al. (2017) and use 20 Pascal classes as seen and 12 Yahoo classes as unseen classes.

NUS-WIDE dataset comprises nearly 270K images with 81 human-annotated categories, in addition to the 925 labels obtained from Flickr user tags. As in Huynh & Elhamifar (2020); Zhang et al. (2016), the 925 and 81 labels are used as seen and unseen classes, respectively.

Open Images (v4) is a large-scale dataset consisting of 9 million training images along with 41,620 and 125,456 validation and testing images, respectively. The scale of Open Images is larger than other multi-label datasets, such as NUS-WIDE and MS COCO. This dataset is partially annotated with human labels and machine-generated labels. Here, 7,186 labels, having at least 100 images in training, are selected as seen classes. The most frequent 400 test labels, which are not present in the training data, are selected as unseen classes, as in Huynh & Elhamifar (2020).

Microsoft COCO (MS-COCO) is a large-scale dataset for object detection, segmentation, and image captioning. We follow the 2014 challenge for data split (i.e., 82783 and 40504 images for training and testing, respectively), including 80 distinct object tags. After discarding images without any labels, we split the training set into 78081 training images and 4000 validation images, and the test set is with 40137 images. We extract and fix image features extracted from layers "7, 33" of ResNet-50 He et al. (2016) for all the baselines considered in our experiments.

A.9 BASELINES:

SSVM is a large margin classifier with a variable margin relying on a structured loss function ℓ. The objective function is defined as follows:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \lambda||\theta||_2 + \sum_n \xi_n \tag{21}$$

$$\text{s.t. } \theta^\top(\phi(y_n^*, \vec{x}_n) - \phi(y, \vec{x}_n)) \geq \ell(y_n^*, y) - \xi_n \quad \forall\, y,$$

where θ is the weight vector (learning parameter), the feature function is represented as ϕ, ℓ is the loss function, and ξ is the slack variable. The SSVM objective function employs an iterative constraint generation strategy to accelerate the learning process by adding a few constraints per iteration (instead of the entire constraint set defined by each label $y \in \mathcal{Y}$). For all experiments, we employ 10-fold cross-validation on the train data to set the value of C.

Costa Mensink et al. (2014) addresses the ZSL problem by performing classification in the original feature space with directly trained classifiers. This method relies on the sample transfer strategy from seen classes to unseen classes based on their transferability and diversity. We mirror the implementation details of Mensink et al. (2014) and run our experiments.

ConSE This method constructs an image embedding system from any existing multi-label image classifier and a semantic word embedding model, which contains the n class labels in its vocabulary Norouzi et al. (2013). We implemented this method following the explanation in Norouzi et al. (2013).

TAEP Ye & Guo (2019) This approach presents a transfer-aware label embedding method for multi-label zero-shot image classification. It projects both labels and images into the same semantic space to rank the similarity scores of the images with positive and negative labels under a max-margin learning framework. Furthermore, it employs label embedding projection with a transfer-aware regularization objective to achieve a promising inter-label relation for information adaptation Ye & Guo (2019). TAEP-C is an extended version of TAEP which uses Flicker Image Hit-Count as additional information. We implemented this method following the explanation in Ye & Guo (2019).

Fast0tag Zhang et al. (2016) the vector offsets in the word vector space. It develops a model to solve image tagging (multi-label classification) by estimating input images' principal directions. We build upon the code provided by Chen et al. (2013) and implement Fast0tag.

LabelEM Akata et al. (2015a) views attribute-based image classification as a label-embedding problem where each class is embedded in the space of attribute vectors. It introduces a function that measures the compatibility between an image and a label embedding. The parameters of this function, given an image, are learned on a training set of labeled samples to locate the correct classes rank higher than the incorrect ones. We follow Akata et al. (2015a) and implement LabelEM.

DeViSE Frome et al. (2013) presents a deep visual-semantic embedding model trained to identify visual objects leveraging both labeled image data and semantic information gleaned from unannotated text. We follow Frome et al. (2013) and implement DeViSE.

GEN Gupta et al. (2021) explores the problem of multi-label feature synthesis in the zero-shot setting. It presents three different fusion approaches (ALF, FLF, and CLF) to synthesize multi-label features. ALF synthesizes features by integrating class-specific attribute embeddings at the generator input. FLF synthesizes features from class-specific embeddings individually and integrates them in feature space. CLF combines the advantages of both ALF and FLF using each individual-level feature and attends to the bi-level context. Consequently, individual-level features adapt themselves producing enriched synthesized features that are pooled to achieve the final output. Lastly, it integrates the fusion approaches in two generative architectures. We have followed the implementation code provided by the GEN authors and reported the experimental results.

LESAHuynh & Elhamifar (2020) presents that designing an attention mechanism for recognizing multiple seen and unseen labels in an image is a non-trivial task as there is no training signal to localize unseen labels and an image only contains a few present labels that need attention out of thousands of possible labels. Hence, the LESA authors let the unseen labels select among a set of shared attentions that are trained to be label-agnostic and focus on only relevant/foreground regions through their loss instead of generating attention for unseen labels which have unknown behaviors and could focus on irrelevant regions due to the lack of any training sample. This classifier learns a compatibility function to distinguish labels based on the selected attention. The loss function consists of three components guiding the attention to focus on diverse and relevant image regions while utilizing all attention features. We follow Huynh & Elhamifar (2020) and implement LESA and One Attention per Cluster.

BiAM Narayan et al. (2021) presents an approach towards region-based discriminability maintaining multi-label zero-shot classification. This method keeps the spatial resolution to maintain region-level characteristics and employs a bi-level attention module (BiAM) to enrich the features by comprising both region and scene context information. The enriched region-level features are then mapped to the class semantics, and only their class predictions are spatially pooled to obtain image-level predictions, thereby keeping the multi-class features disentangled. We run our experiments employing the code provided by the authors.

For other baselines like, **SEKG** Wang et al. (2018), **EXEM** Changpinyo et al. (2017), **DMP** Fu et al. (2015), , **LatEM** Xian et al. (2016), we follow their paper and implementation and report the experimental results.

A.10 MORE EXPLANATION ABOUT MIN-MAX MULTI-LABEL ZSL

- **Exact loss and probabilistic method advantage:** A classifier can learn better when the objective is well defined. Leveraging exact loss helps define the objective function more accurately, while other methods like SVM and Logistic regression approximate the loss.

No Over-fitting: leveraging exact loss does not lead to overfitting due to the well-defined constraints and min-max distributions. In the training phase, different probabilistic distributions are assigned to the possible labeling of the nodes by the maximizer and the minimizer to solve this min-max game as follows (Eq.5):

$$\begin{aligned}
&\min_{P_{mini}} \max_{P_{maxi}} \mathbb{E}_{x \sim P_{Data}; y_{maxi}|x \sim P_{maxi}; y_{mini}|x \sim P_{mini}} \left[\text{loss}(Y_{mini}, Y_{maxi})\right] \\
&\text{such that: } \mathbb{E}_{x \sim P_{Data}; y_{maxi}|x \sim P_{maxi}} \left[\phi(X, Y_{maxi})\right] = \tilde{\mathbf{c}}.
\end{aligned}$$

These probabilistic assignments (P_{maxi} and P_{mini}) have two key advantages: 1) Considering eq. 5, they give different weights (based on closeness to true labels) to the labels and their features. Therefore, the learning parameter is adjusted considering these probabilities. 2) These probabilistic assignments stimulate the augmentation process by assigning different probabilities to possible labelings and their features. Therefore, a labeling (and its related feature representation) may get different probability distributions which are like augmentation. Therefore, there is no over-fitting.

- **Explaining Our Method Using A Simple Example:** For simplicity, consider a graph with three nodes(sea, **fish**, and tiger) and edges (pair-wise features). Our algorithm learns to make min-cut to recognize present classes in the images. We use our trained classifier in the ZSL setting with classes like the sea, **dolphin**, and tiger. Dolphin and fish share more similar feature representations considering pairwise and taxonomy feature representations. Therefore, the edges (dolphin-sea), (dolphin-tiger) get similar feature representation to (fish-sea), (fish-tiger). It stimulates the same cut used for sea, fish, and tiger classes. Thus, making a correct min-cut by our classifier.