Convex Adversarial Collective Classification
Ali Torkaman and Daniel Lowd
University of Oregon

Adversarial Collective Classification

Goal: Learn to robustly label a set of related objects in the presence of adversarial manipulation.
Applications:
• Adversarial Manipulation: Collective classification problems in which the test data is manipulated by an active adversary to maximize the misclassification error. Examples: web spam, counter-terrorism, auction fraud, etc.

Concept Definition: Collective classification problems in which distribution of test data has diverged from the distribution of data at train time. For example, when classifying blogs, tweets, or news articles, the topics being discussed will vary over time.

Scenario:
• Political Blogs: 10%
• Reuters: 20%
• Experimental Setup

Datasets
• Synthetic: 20%
• In the next sections of the poster we will use the following notation:

Motivation and Overview

An associative Markov network (AMN) is a Markov network where linked nodes are more likely to have the same label. [Taskar et al., 2004]

Learner’s Goal: Select \( \mathbf{w} \) to maximize the margin between true labeling and alternate labeling:

\[
\min \quad \frac{1}{2} \sum_{i \in V} \mathbf{w}_i^2 + C_L \quad \text{s.t.} \quad \mathbf{\Delta} (\mathbf{x}, \mathbf{x}_A) \leq 0 \quad (2)
\]

Where \( \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \) is the number of misclassified nodes.

Good News: For some functions being bilinear and convex (i.e., \( \text{score}(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \mathbf{x} \mathbf{y} \) ), we can derandomize the problem.


Notation

In the next sections of the poster we will use the following notation:

Convex Adversarial Collective Classification

Goal: Learn to jointly predict the labels of the nodes in an AMN, while being aware of possible existence of active adversary at test time.

What can an adversary do?

• Adversary’s Weakness:
  • It has a budget \( D \) for the maximum number of features that it can change. For \( \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \) being the difference measure between the true features \( \mathbf{x} \) and the features after adversarial manipulation \( \mathbf{x}_A \), we have \( \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \leq D \).

• Adversary’s Inference:
  • Given the parameters \( \mathbf{w} \), the adversary can choose \( \mathbf{x}_A \) such that the alternate labeling receives a high score, making it hard for the classifier to predict the correct joint labels, plus getting a reward when the alternate labeling is more different from the true labeling.

The adversary can achieve this by solving the following non-convex program:

\[
\max_{\mathbf{x}, \mathbf{y}} \text{score}(\mathbf{x}, \mathbf{y}, \mathbf{w}) - \text{score}(\mathbf{x}, \mathbf{y}_A, \mathbf{w}) + \mathbf{\Delta}(\mathbf{y}, \mathbf{y}_A) \quad \text{s.t.} \quad \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \leq D
\]

What can the learner do?

• The learner should be robust against rational adversaries; this can be achieved by introducing an adversarially constrained large margin SVM:

\[
\min \quad \frac{1}{2} \sum_{i \in V} \mathbf{w}_i^2 + C_E \quad \text{s.t.} \quad \mathbf{\Delta} (\mathbf{x}, \mathbf{x}_A) \leq 0 \quad (3)
\]

Can we solve them?

• With the score function being linear in each of the variables (i.e., \( \text{score}(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \mathbf{x} \mathbf{y} \) ), both of the programs (2) and (3) are non-convex.
  • Since the program (3) is the same as the inner maximization in program (2), we can use the same trick for solving both of the problems. The procedure is as follows:

1. Convert the trilinear form in the score function that has both \( x \) and \( y \) to bilinear form, by introducing a dummy matrix variable \( \mathbf{z} \).
2. Add necessary linear constraints on \( \mathbf{z} \) to make \( \text{score}(\mathbf{x}, \mathbf{y}, \mathbf{w}) = \mathbf{w}^T \mathbf{x} \mathbf{y} \) over the whole program.
3. By substituting the dual of the resulted linear program with the inner maximization in equation (2), the bilinearity will be removed and final program will become a convex standard quadratic program that can be solved efficiently.

Theorem: Equation (2) has an integral solution for binary valued \( x \) and \( y \).

Associative Markov Networks

• An associative Markov network (AMN) is a Markov network where linked nodes are more likely to have the same label. [Taskar et al., 2004]

Learner’s Goal: Select \( \mathbf{w} \) to maximize the margin between true labeling and alternate labeling:

\[
\min \quad \frac{1}{2} \sum_{i \in V} \mathbf{w}_i^2 + C_L \quad \text{s.t.} \quad \mathbf{\Delta} (\mathbf{x}, \mathbf{x}_A) \leq 0 \quad (2)
\]

Where \( \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \) is the number of misclassified nodes.

Good News: For some function being bilinear in \( x \) and \( y \) and convex (i.e., \( \text{score}(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \mathbf{x} \mathbf{y} \) ), we can derandomize the non-convex bilinear mathematical program in (2) to a convex standard QP, by substituting the inner maximization linear program with its dual.

What can an adversary do?

• Adversary’s Weakness:
  • It has a budget \( D \) for the maximum number of features that it can change. For \( \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \) being the difference measure between the true features \( \mathbf{x} \) and the features after adversarial manipulation \( \mathbf{x}_A \), we always have \( \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \leq D \).

• Adversary’s Inference:
  • Given the parameters \( \mathbf{w} \), the adversary can choose \( \mathbf{x}_A \) such that the alternate labeling receives a high score, making it hard for the classifier to predict the correct joint labels, plus getting a reward when the alternate labeling is more different from the true labeling.

The adversary can achieve this by solving the following non-convex program:

\[
\max_{\mathbf{x}, \mathbf{y}} \text{score}(\mathbf{x}, \mathbf{y}, \mathbf{w}) - \text{score}(\mathbf{x}, \mathbf{y}_A, \mathbf{w}) + \mathbf{\Delta}(\mathbf{y}, \mathbf{y}_A) \quad \text{s.t.} \quad \mathbf{\Delta}(\mathbf{x}, \mathbf{x}_A) \leq D
\]

What can the learner do?

• The learner should be robust against rational adversaries; this can be achieved by introducing an adversarially constrained large margin SVM:

\[
\min \quad \frac{1}{2} \sum_{i \in V} \mathbf{w}_i^2 + C_E \quad \text{s.t.} \quad \mathbf{\Delta} (\mathbf{x}, \mathbf{x}_A) \leq 0 \quad (3)
\]

Can we solve them?

• With the score function being linear in each of the variables (i.e., \( \text{score}(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \mathbf{x} \mathbf{y} \) ), both of the programs (2) and (3) are non-convex.
  • Since the program (3) is the same as the inner maximization in program (2), we can use the same trick for solving both of the problems. The procedure is as follows:

1. Convert the trilinear form in the score function that has both \( x \) and \( y \) to bilinear form, by introducing a dummy matrix variable \( \mathbf{z} \).
2. Add necessary linear constraints on \( \mathbf{z} \) to make \( \text{score}(\mathbf{x}, \mathbf{y}, \mathbf{w}) = \mathbf{w}^T \mathbf{x} \mathbf{y} \) over the whole program.
3. By substituting the dual of the resulted linear program with the inner maximization in equation (2), the bilinearity will be removed and final program will become a convex standard quadratic program that can be solved efficiently.

Theorem: Equation (2) has an integral solution for binary valued \( x \) and \( y \).

Classifer and Adversary’s inference

• Both the classifier and adversary’s inference problems are linear programs.
• Equation (2) is the adversary’s linear program.
• Classifier’s inference for predicting the joint labeling of nodes, is the same as in an AMN.

Experiments

• Experimental Setup
  • Baseline methods: AMN [Taskar et al., 2004] and SVM.
  • Robust methods: SVMInvar (Teo et al., 2008) (Baseline) and CACC (Our method).
• Parameter C for all methods and adversary’s train budget D, are tune with 0%, 10%, and 20% of adversarial manipulation strength at the tuning data.
• Datasets
  • Synthetic: 10 random graphs, each with 100 nodes (half positive and half negative labels) and 30 Boolean features. Nodes are more likely to link to the ones that have the same label, and half of the nodes were only recognizable by their links.
  • Political Blogs, collected by [Adamic et al. 2005]. We extended this dataset by crawling the blogs at different times and cleaning dead pages manually. In this dataset, we observe some concept drift at different times.
  • Reuters: ModApte split of the Reuters-21578 corpus. Four classes: crude, grain, trade, and money-fx are selected.

Comparison with baselines

Synthetic: 0%  Synthentic: 10%  Synthetic: 20%

Political Blogs: 0%  Political Blogs: 10%  Political Blogs: 20%  Reuters: 0%  Reuters: 10%  Reuters: 20%

Conclusion and future work

• Robustness combined with the ability to reason about interrelated objects
• Representation of the adversarial learning task as a bilinear quadratic Stackelberg game

Future work:
Extend our method to learn adversarially regularized variants of non-associative relational models, also scale to large size problems where many of which are semi-supervised.