A Content-Oriented Approach to Vertex Nomination

Ashequl Qadir, Eleanor Wong, Alex Clemmer

Proposal for Final Project
Machine Learning (CS 5350/6530)

October 18, 2011

Abstract

We are given a graph $G = (V, E)$. Commonly it is the case that we have verified knowledge of some attribute $\vec{x} \in \mathbb{R}^D$ for only some fraction of the vertices $\mathcal{V} \subset V$, and that it is either extremely expensive or impossible to directly verify $\vec{x}$ for any particular new vertex $v \in V$. The space of methods that produce a list of vertices $\mathcal{Q} \subseteq V$ that are likely to have $\vec{x}$ is called vertex nomination [Coppersmith & Priebe, 2011].

We will build off existing work to apply this framework to the Enron Email Corpus. This corpus is somewhat unique in that it consists of emails siezed by the Justice Department in their 2002 investigation of the Enron Corporation, and contains some emails that provide evidence that the correspondents were involved in criminal activity. We a propose a supervised learning approach that will leverage the data in these emails to "nominate" a list of vertices that are likely to have this quality. Specifically we propose building a variable-length n-gram language model using the Prediction by Partial Match (PPM) algorithm [Cleary, Witten, 1990]. We will use PPM to determine the probability that some blob of content indicates criminal activity, and subsequently, we will use a mapping $f : [0, 1] \to [0, 1]$ of email probabilities to vertex probabilities to construct a list of the k vertices that are most likely to be "evil".

1 Introduction

In the context of the Enron corpus, the graph $G = (V, E)$ each $v \in V$ represents an email address, and each $e \in E$ represents the content of a specific email. Note that an email can be addressed to many people, in which case there would be corresponding edges between different vertices that contain the same content.

In the context of the problem, we will think of the $\mathcal{V}$ as the set of vertices which we know were indicted for criminal activity. Such vertices are said to be *red*; in contrast, vertices that are not known to be indicted for criminal activity are said to be *nonred*. It is particularly important to note that nonred vertices are not necessarily not criminals: rather, they are not *known to be criminals*. In other words, if we return a list of k vertices, and the list contains none of the vertices that we know to be red, our list may not be wrong, since we do not know that nonred vertices are not actually criminals.

It is also important to note that the vertex color pertains specifically to the context of the experiment. This means that it is possible to, say, "hide" some of the red vertices for training purposes, and then evaluate success on some metric (*e.g.*, precision of results, which is defined to be $\frac{\text{number of correct answers}}{\text{number of possible correct answers}}$).

2 Previous Work

Previous approaches to vertex nomination in the context of the Enron corpus [Coppersmith & Priebe, 2011] have ignored the email-analytic approach, instead focusing exclusively on graph-oriented features like graph topology to nominate their list of k vertices.

Our proposal differs in that we will mainly be concerned with nomination specifically from the content-analytic approach. It is notably possible to augment the content- and context-based approaches, and if we have time, we will do this, but it will not be our priority.

Previous approaches evaluate success using mean precision (MP) over a set of cross-validated experiments. Specifically, a given experiment will "hide" some of the red vertices, and evaluate precision for that experiment as $\frac{\text{number of red vertices returned}}{\text{number of red vertices in total}}$. Because the experiments are jacknifed, we end up with 100 such numbers; the evaluation metric is the mean of these precisions.

Whether this is the right thing to do is entirely debateable. One case where it clearly fails is if our selected vertices are only moderately "evil", then a perfect algorithm would return a top-k list that would include only the worst bad guys, which would very likely include either very few known red vertices, or none at all. In this case, the MP would be definitively wrong, since it would be 0% or close to 0% every time.

We probably do not have time to fix this problem. Instead, we will likely just use their metric, which also has the advantage of making our results directly comparable to theirs. If we come across a vastly better way to evaluate, we will use that, but this is not our expectation for this project.

3 Procedure

We begin by training a model. We will use a composite model—two models that each output a probability individually, which we then normalize and return. One of the sub-models will be trained on content that occurs between two or more red vertices, which in other words is content that is most likely to be indicative of "evil" behavior. The other of the sub-models will use content that occurs between all non-red vertices.

We will train our model at the sentence level, not the email level, because our experiments so far have demonstrated this to be more effective. PPM models based on variable-length n-grams; in this context, our grams will be *characters*, which makes a 3-gram the characters in length. The reason, again, is that this has proven to be much more effective in our experiments so far.

We begin training by taking some set of training vertices, partitioning them up according to the division between the sub-models, and then training on respective relevant content. Essentially, the process begins by taking all of the respective emails, and feeding them into the model sentence by sentence.

Because we are using a composite model, scoring is slightly tricky. To score one sentence, we supply the sentence directly to each of the sub-models. Both of these will output a log prob. The first task is to turn this into a probability, which we do by raising it as a power over 2, *e.g.*, probability $= 2^{\log \text{prob}}$.

This so far will cause our score to prefer shorter sentences, so we will normalize this with respect to sentence length. Since our n-grams are characters, we do this simply by dividing by total length of sentence in characters.

Because this is a composite model, we want to balance the assertion that a given sentence indicates "evil" behavior (*i.e.*, prob_{red} with the assertion that it is not prob_{nonred}. The way to do this is to normalize these probabilities, so that $\text{prob}_{red} + \text{prob}_{nonred} = 1$. For example, $\text{prob}_{red} = \frac{\text{prob}_{red}}{\text{prob}_{red}+\text{prob}_{nonred}}$. We then return prob_{red}, which is the probability that a given sentence indicates "evil" behavior.

So far we have a way to "score" sentences, so the question is how we apply these scores to a vertex. We begin by picking any vertex v that we'd like to score. We then take all of the emails that this vertex is involved with, and feed each of the t sentences into the model on a sentence-by-sentence basis. This results in t probabilities that correspond to the "evilness" of each of the t sentences. This is representable as $\vec{p} \in [0,1]^t$.

Here we have a dilemma: we can define the score of a given vertex as $\text{score}_v = \max(\vec{p})$, but because we are basing our score on one datapoint and one datapoint only, we will get a high anomaly rate. We can define $\text{score}_v = \text{average}(\vec{p})$, but this will wash out red vertices that have a few really bad conversations, but lots more non-bad conversations.

Our solution is to take the average of the top m sentences. The intuitive reason is that we want to see roughly how bad the conversations that are most bad, are. By taking the average of some top m, we can avoid the washout and the anomaly rate simultaneously.

So at this point every vertex will have a probability. Since we are using the (vert flawed) metric of success mentioned above, we will subsequently return the top k, sorted by this probability. Success then becomes the MP of these cross-validated experiments.