Verb Clustering Project Write-up

Kyle Sargent

August 10, 2017

Abstract

This document is intended to be a readable discussion of my findings regarding the relationship between VerbNet and GLoVe. Comments welcome!

1 Predicting VerbNet frames from GLoVe vectors

In this section, I will attempt to demonstrate for my lab supervisors that GLoVe vectors for verbs encode statistically significant syntax information. Where possible, the simplest models and evaluation benchmarks have been chosen to provide the most convincing proof. First, some background is in order.

1.1 Introduction and Background

Levin (1993) developed the notion of a verb class. This classification was extended and enriched by Verbnet, a database of about 8000 verb/class instances. The website of VerbNet describes a verb class:

> "Each verb class in VN is completely described by thematic roles, selectional restrictions on the arguments, and frames consisting of a syntactic description and semantic predicates with a temporal function, in a manner similar to the event decomposition of Moens and Steedman (1988)."

We are interested in demonstrating that GLoVe encodes syntax information; thus, the most natural thing to try and predict are the frames a verb fits into. It is worth noting that in VerbNet, whether or not a verb fits in a frame is true or false, with no gray area - in practice, such judgments by humans are much fuzzier.

Any verb in VerbNet belongs to some number of classes, and each class expresses some number of the 291 frames. Suppose we number the frames $f_1 ... f_{291}$. The notion of a "binary frame vector" associated to a verb is as follows: Suppose a verb vhas associated binary frame vector v, then we assign the following value to entry i of the frame vector:

$$v_i = 1\text{if } v \text{ belongs to frame } i \text{ in any of its class instantiations, and 0 otherwise}$$

On the other hand, for most of the verbs in VerbNet, we have GLoVe vectors. For this project, I wanted to make use of distributional information from Wikipedia and Twitter - thus, GLoVe vectors from both were concatenated to form a 400-dimensional vector of real numbers that encodes its distributional information.

1.2 Summary of the Prediction Task

We can reduce from the problem of recovering verb syntax from GLoVe to learning a good approximate mapping from the 400-dimensional GLoVe space to the 291-dimensional space of binary frame vectors. In the field of machine learning, this is known as a multi-label classification problem.

We'll randomly shuffle the set of verbs for which we have both GLoVe vectors and frame vectors into a training set of 2949 verbs, and a validation set of 737 verbs, i.e. the standard 80/20 training/validation split for most machine learning problems. By learning a mapping from glove vectors to frames on the training data and checking its precision (see part 2) on the validation data, we will attempt to demonstrate that we can in fact successfully recover syntax information from GLoVe.

One of the simplest possible models was chosen for classification: a logistic regression model. Since a logistic regression model is a binary classifier, and we are interested in 291 classification problems, we simply concatenate and separately train 291 independent logistic classifiers to predict the frame vector of a verb from its GLoVe vector.

An advantage of a logistic regression model is that its transparency will allow us to directly examine the weights on each dimension of GLoVe - in short, it may be able to find which dimensions of GLoVe are most diagnostic, for syntax purposes.

To ensure robust predictions, positive training examples in both training and testing sets, and to simplify the problem of disentangling the data into training and testing sets such that this latter condition is satisfied, all frames with 5 or fewer positive training examples have been excluded from our prediction task. Since there are 232 frames with more than 5 positive training examples, we restrict our investigation to these frames (for the time being).

1.3 Discussion of Results

Since we are evaluating hundreds of independent binary classifiers, the Area Under the Receiver Operating Curve (AUROC) seems a fair benchmark for our model. The receiver operating curve of a binary classifier is essentially the true positive rate of the classifier, plotted against its false positive rate. That is, the probability it will correctly classify a positive example as a function of the probability it will incorrectly classify a negative example. When these points are joined they form a curve, and the area under it, or AUROC, measures the effectiveness of a binary classifier by giving a number between .5 and 1.0 that measures the strength of the tradeoff between false positives and true positives - at .5, a model is no better than a random classifier. At 1.0, the model classifies all training examples perfectly.

After fitting the 232 logistic regressions on the training data, we now consider their performance on the testing data. We can plot the receiver operating curves atop one another as below. They appear vary widely, but there is no mistaking that the average predictive strength is well above a random classifier. However, some spuriously trained models do quite poorly - take a look at the curves that fall below the random classifier benchmark line $y = x$.

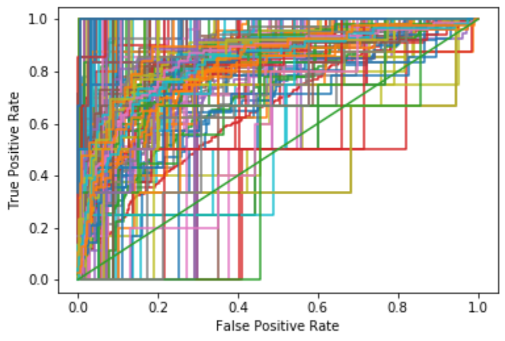

Figure 1: 232 frames with minimum 5 training examples, AUC average = .8431

It is easy to see that, on average, predicting frames from GLoVe vectors using a logistic regression classifier does better than random chance. But there is a lot of variance in how good the model is for different frames. Let's see what happens when we raise the threshold for training examples.

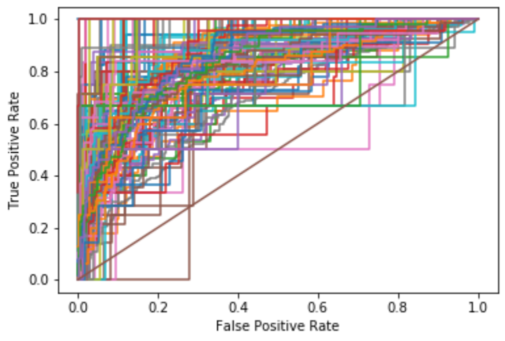

Figure 2: 155 frames with minimum 15 training examples, AUC average = .8421

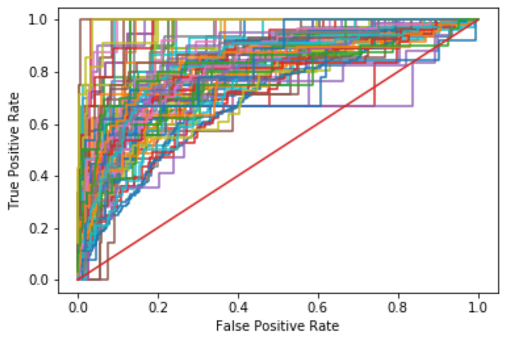

Figure 3: 93 frames with minimum 30 training examples, AUC average = .8409

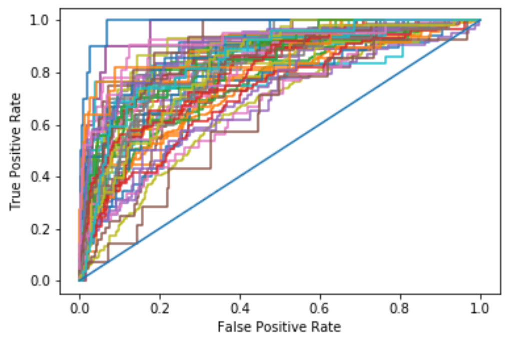

Figure 4: 60 frames with minimum 50 training examples, AUC average = .8111

It's clear from the above plots that model stability and performance goes up as a function of the number of training examples. What's encouraging is that AUC seems to stabilize as well. Let's take a look at a scatterplot of AUC against the number of training examples.

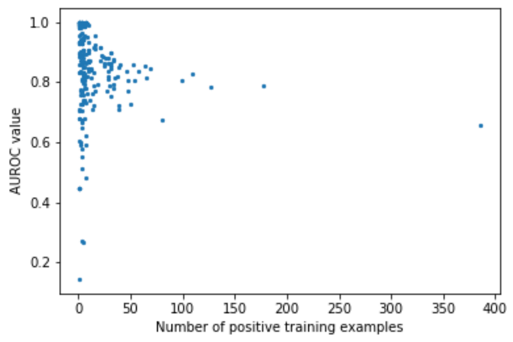

Figure 5:

As training examples (the X-axis) increases, the spread in AUC tends to decrease, all the while stabilizing at around .75, confirming our hypothesis.

It seems clear that our model is predicting syntax reasonably well from GLoVe vectors. How can we be sure we have not overfitted a model? For one, the training data was thrown away after model fitting, and all the results you see above are on the test set. This alone is reason to believe we have not overfitted to our data.

Suppose we scrambled the assignment of verbs to frames. Our model could still pick up on the distributional information of frames (e.g. this frame occurs 30% of the time, so with a false positive tolerance of 70% or more we should predict it always), but the GLoVe vector observations for a verb would be meaningless. How does such a model do? Let us examine their collective ROC plots.

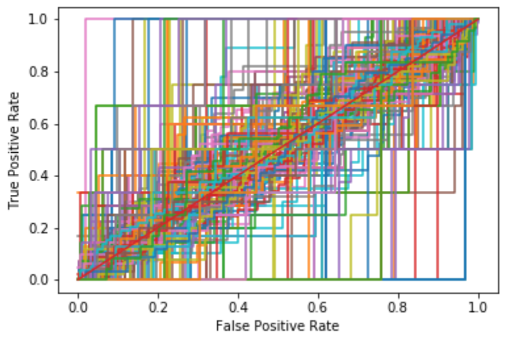

Figure 6:

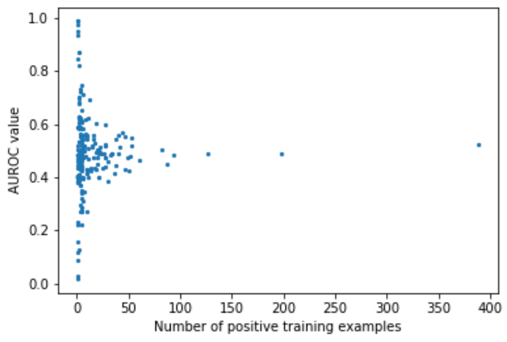

Figure 7:

These models perform no better than random chance. Indeed, it's clear in this case that

$$\lim_{\#\text{training examples}\to\infty} AUC = .5$$

Which is as we expect for a collection of fundamentally random (uninformed) classifiers. This is good because it tells us GLoVe vectors give us frame information when (and only when) this information is correctly paired, which is another confirmation of our hypothesis.

2 Diagnostic Verbs and Diagnostic Frames

The following section describes my attempt to answer the following question: Which verbs and frames are the most "diagnostic?" Here, we take diagnostic to mean "explanatory" in the following sense - how can we pick a collection of frames and verbs that best represent their resepective datasets? In both cases, I have decided to use clustering to ensure maximal coverage of our dataset.

2.1 Diagnostic Frame Clustering

Unfortunately, clustering requires the space of frames to be endowed with a distance metric. There is not a canonical notion of "distance" between two frames, so I have decided that frames which occur in largely the same verbs should be "close" and frames which don't occur in the same verbs should be "far." There is a mathematically nice way for defining a distance metric on the space of finite sets - *Jaccard distance*. This distance will capture the above, naive intuition about frame distance.

Consider two frames f_1 and f_2 and the set of verbs in which they appear, V_1 and V_2. The *Jaccard* distance between V_1 and V_2 is defined

$$d_J(V_1, V_2) = 1 - \frac{|V_1 \cap V_2|}{|V_1 \cup V_2|}$$

That this constitutes a formal distance metric on the space of finite sets is out of scope, but this fact is required to justify clustering. Nevertheless, it is easy to see that dissimilar frames will be "far apart." If the set of verbs in which both frame f_1 and f_2 appear is empty, $d_J = 1 - \frac{0}{|V_1 \cup V_2|} = 1$. Likewise if they appear in only the same verbs, their distance is zero (for all practical purposes, the frames are identical)

With a distance metric, we can run spectral clustering (preferred over standard K-means clustering because it can detect non-Euclidean clusters like long, thin agglomerations) with as many clusters as we like.

2.2 Diagnostic Verb Clustering

Due to computational limitations, spectral clustering is infeasible for the several thousand verbs for which I have GLoVe vectors, even with PCA to reduce their dimensionality. Thus, simple K-means clustering was used to cluster verbs.

2.3 Diagnostic Table

The purpose of determining a set of diagnostic frames and verbs is to construct a table T that has columns labeled with frames and rows labeled with verbs, such that the entry in row v and column f is "True" if verb v fits into frame f, and false otherwise. With this table, validation of VerbNet and our logistic regression method on humans can be simplified.

We are motivated to produce a suitably rich table for human testing. On average, a verb fits into about 4.4 frames, so having no priors we could expect about $4.4/291 \approx 1.5\%$ of the entries in our table to be actual instances of a frame fitting into a verb, which is not ideal. We would like to push percentage p_{true} higher so humans can maintain their focus in the verb/frame matching task.

This is accomplished via the following randomized procedure after clustering the frames into n_f clusters and the verbs into n_v clusters. First, we pick a random frame from each frame cluster and a random verb from each verb cluster, to furnish a concrete set of n_f diagnostic frames and a concrete set of n_v diagnostic verbs. We then construct an $n_v \times n_f$ table as described above – with rows labeled by verb and columns labeled by frames. Finally, we drop the $n_v/2$ sparsest rows and the $n_f/2$ sparsest columns (in that order) to produce a smaller but richer $n_v/2 \times n_f/2$ table.

With the intent of producing a 50x25 table, we can start with 100 verb clusters and 50 frame clusters and, after clustering, run the procedure above until we get a suitably rich table, for example with $p_{true} \approx 10\%$.