Neural Stack Writeup

November 24, 2017

Contents

1 Structure

1.1 Introduction

What did you do? Why?

1.2 Methods

In this section, we describe the dataset used, experimental setup and details of models evaluated.

[*Dataset description like earlier paper(s)*]

Unlike previous work, we don't rely on the performance of component models for estimating a set of *weights*. Instead, we use the predictions (probability distributions) from the components as input to the ensemble model and predict directly the output for the wILI prediction problem, skipping any intermediate weight estimation.

1.2.1 Component model inputs / outputs

[*Something similar to earlier papers*]

1.2.2 Ensemble model inputs

The ensemble models *merge* probability distributions from input component models and predict probability distributions, in the same space, as output. During training, the mean categorical crossentropy loss between the output distribution and the actual wILI value, as one-hot distribution, is minimized. This loss minimization is equivalent to maximizing the log score as described by CDC [*Elaborate more on the CDC loss*].

Other than the probability distributions, the ensemble models also have information about the current week (the week they are merging probabilities for) of the season. Since week goes from 1 to 52/53, to preserve continuity, we encode each week (w) in a vector $\bar{w}$ of two elements in a sinusoidal representation as follows:

$$\bar{w} = [\sin(\frac{2\pi w}{n}), \cos(\frac{2\pi w}{n})]$$

Where n is the total number of weeks (52/53) in the season year. For each neural network model, we create a *with-week* variant which takes in $\bar{w}$ as one of its input.

1.2.3 Models

We evaluate two neural network models for the stacking task. The first model (mixture density network) works by approximating the input probability distributions using a gaussian and the output as a mixture of gaussians. The second model (convolutional neural network) works directly on the probability distributions (as vector of bin values) from components as input and returns a vector of values as probability distribution as output. Both models are described below:

1. **Mixture density network**

   A mixture density network [*CITE*] is a simple feed forward neural network which outputs parameters for a mixture of distributions. The loss function here is the crossentropy loss between the mixture of distributions generated by the network and one-hot representation of the truth.

This model *assumes* a gaussian distribution input from the component models. It takes in the mean and standard deviation of the distribution from each of the component models and returns a mixture of gaussians by outputting a set of means (μ_i), standard deviations (σ_i) and weights (w_i) for each distribution in the mixture. The final distribution for a network outputting n mixtures is then given by:

$$F(x) = \sum_{i=1}^{n} w_i f(x, \mu_i, \sigma_i^2) \tag{1}$$

Where $f(x, \mu_i, \sigma_i^2)$ represents a gaussian with mean μ_i and variance σ_i^2. Figure [@fig:mdn] shows the structure of a mixture density model (with weeks) [*REFER TO GITHUB*]

2. **Convolutional neural network**

   This model puts less assumptions on the input and output distributions and uses a set of 1-dimensional convolutional layers over the discrete input distributions. As the output, it outputs the complete discrete probability distribution vector. Figure [@fig:mdn] shows the structure of a convolutional model with weeks [*REFER TO GITHUB*]

1.3 Results

What did you find?

1.4 Discussion

What does it all mean?

1.5 Conclusions

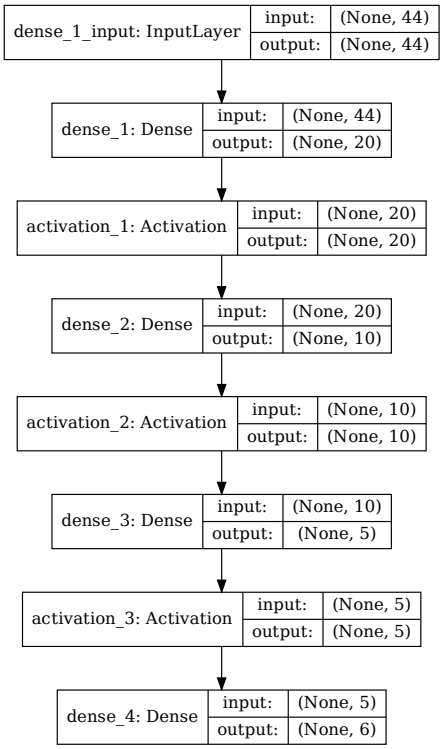

Figure 1: Graph of the mixture density network model. This specific network takes in means and standard deviations of 21 component models (42 inputs) and 2 inputs encoding week. It outputs 6 parameters to be interpreted as weights, means and standard deviations for a mixture of 2 gaussians.

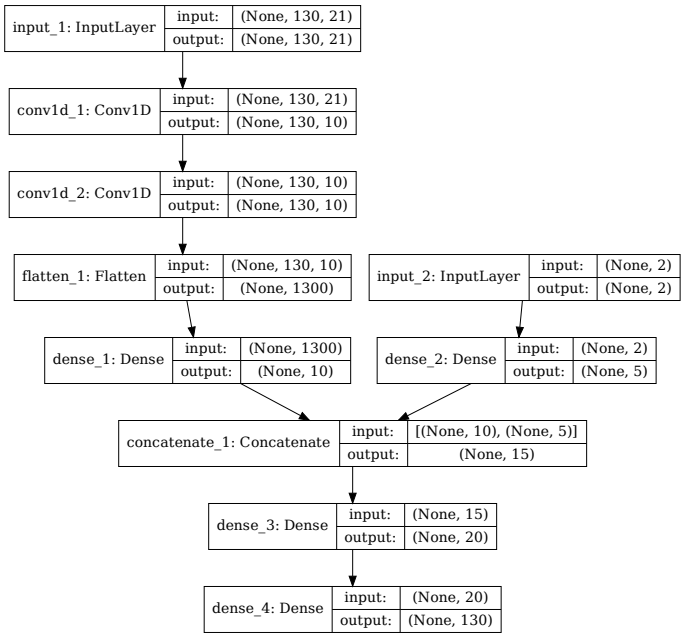

Figure 2: Graph of a convolutional neural model for wili target. The input on the left branch is a set of probability distributions (130 bins) representing wili values for 21 component models. The right branch takes in encoded weeks as vector of size 2. The model finally outputs a probability distribution using 130 bins (same as the component models).