10.5 Recurrent Neural Networks

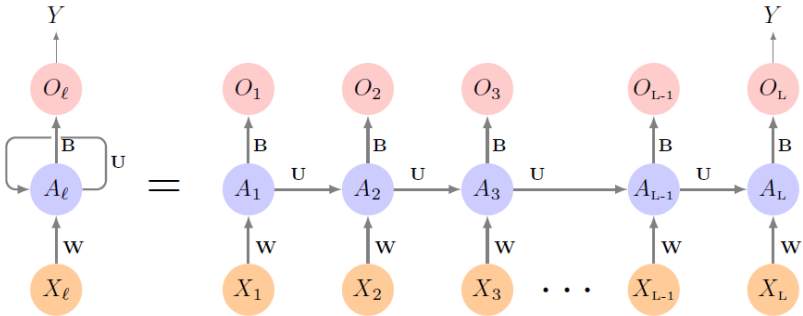
Sequential Data

- Many data sources are sequential in nature, which requires special modeling considerations.
- Examples include
  - Written documents. The sequence and relative positions of words in a document capture the narrative, theme, and tone.
  - Time series such as weather data or financial indices.
  - Recorded speech or music.
  - Handwritten documents.

Recurrent neural network (RNN)

- RNNs build models that take into account the sequential nature of the data and build "memory" of the past.
  - The feature is a sequence of vectors $X = \{X_1, X_2, \ldots X_L\}$
  - The target Y is often a single variable such as `sentiment` or some classification.
  - However, Y can also be a sequence, such as the same document in a different language.

Recurrent neural network (RNN)



- The hidden layer is a sequence of vectors A_l, recieving as input X_l as well as A_{l-1}.
  - A_l produces some output O_l
- The same weights W, U, and B are used as each step (hence *recurrent*).
- The A_l sequence represents an evolving model for the response that is updated as each element X_l is processed.

RNNs in Detail

Suppose $X_l = (X_{l1}, X_{l2}, \ldots, X_{lp})$ has p components and $A_l = A_{l1}, A_{l2}, \ldots, A_{lK}$ has K components.

Then the computation at the kth component of hidden unit A_l is

$$A_{lk} = g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} X_{lj} + \sum_{s=1}^{K} u_{ks} A_{l-1,s}\right)$$

$$O_l = \beta_0 + \sum_{k=1}^{K} \beta_k A_{lk}$$

RNNs in Detail

Often we are concerned only with the prediction O_L at the last unit.

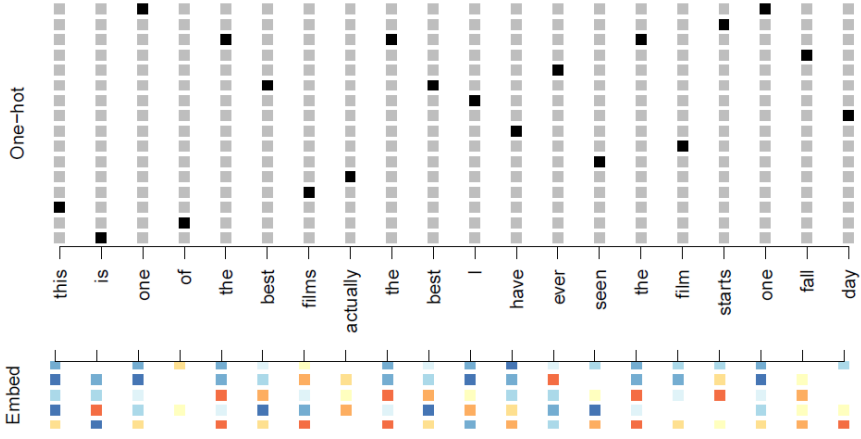
For squared error loss, and n sequence/response pairs, we minimize

$$\sum_{i=1}^{n}(y_i - o_{iL})^2 = \sum_{i=1}^{n}\left\{y_i - \left[\beta_0 + \sum_{k=1}^{K}\beta_k g\left(w_{k0} + \sum_{j=1}^{p} w_{kj}x_{iLj} + \sum_{s=1}^{K} u_{ks}a_{i,L-1,s}\right)\right]\right\}^2$$

IMDb Reviews

- The document feature is a sequence of words $\{W_l\}_1^L$
  - We typically truncate/pad the documents to the same number L of words each (we use $L = 500$).
- Each word W_l is represented as a *one-hot encoded* binary vector X_l (dummy variable) of length $10K$, with all zeros and a single one in the position for that word in the dictionary.
- This results in an extremely sparse feature representation, which does not work well.
- Instead, we use a lower-dimensional pretrained *word embedding* matrix E.
  - This reduces the binary feature vector to a real feature vector of dimension $m << 10K$

Word Embedding

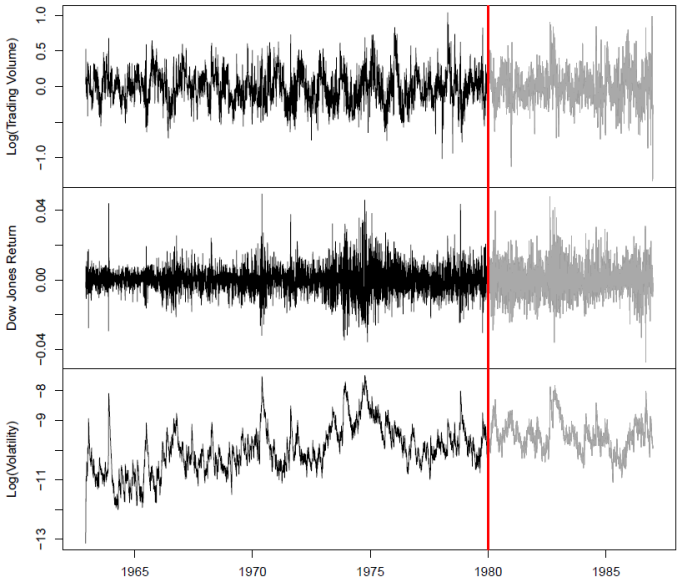


Embeddings pretrained on a very large corpora of documents, using methods similar to principal components.

RNN on IMDb Reviews

- After a lot of work, we get only 76% accuracy.
- However, a more involved RNN with LSTM (*long and short term memory*), does better.
  - Here, A_l recieves input from A_{l-1} (short term memory) and from a version that reaches farther back in time (long term memory).
  - Now we get 87% accuracy, slightly less than the 88% achieved by `glmnet`
- THese data have been used as a benchmark for new RNN architectures.
  - As of 2020, the best reported result was around 95% (and the leaderboard referenced by the textbook does not appear to exist anymore).

Time Series Forecasting



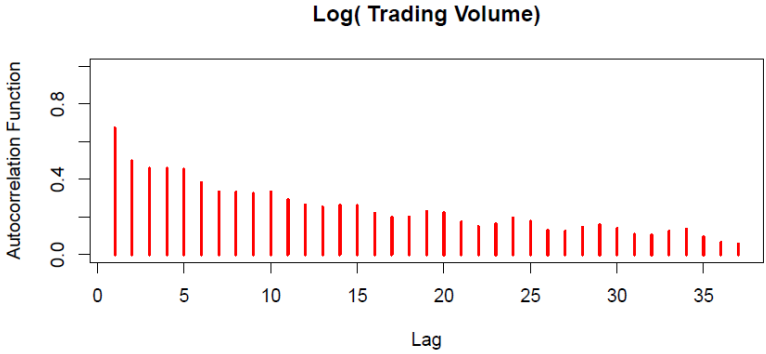
Stock Exchange Data

- The previous figure shows historical trading statistics from NYSE.
- Shown are three daily time series from Dec 3, 1962 through Dec 31, 1986.
  - `Log trading volume`. This is the fraction of all outstanding shares that are traded on that day, relative to a 100-day moving average of part turnover.
  - `Dow Jones return`. The difference between the log of the Dow Jones Industrial Index on consecutive trading days.
  - `Log volatility`. Based on the absolute values of daily price movements.

Stock Exchange Data

Goal: predict `log training volume` tomorrow, given its observed values up to today, as well as those of `Dow Jones return` and `Log volatility`.

Autocorrelation



- The *autocorrelation* at lag l is the correlation of all pairs (v_t, t_{t-l}) that are l trading days apart.
- These sizable correlations give us confidence that past values will be helpful in predicting the future.
- This is a strange prediction problem in that the response v_t is also a feature v_{t-l}.

RNN Forecaster

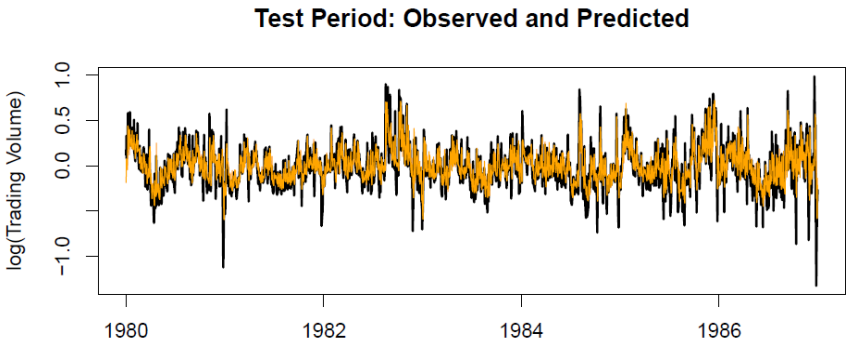
We only have one series of data... how do we set up for an RNN?

- Extract many short input sequences $X = \{X_1, X_2, \ldots, X_L\}$ with a predefined length L known as the *lag*:

$$X_1 = \begin{pmatrix} v_{t-L} \\ r_{t-L} \\ z_{t-L} \end{pmatrix}, X_2 = \begin{pmatrix} v_{t-L+1} \\ r_{t-L+1} \\ z_{t-L+1} \end{pmatrix}, \ldots, X_L = \begin{pmatrix} v_{t-1} \\ r_{t-1} \\ z_{t-1} \end{pmatrix}, \text{ and } Y = v_t$$

- Here, $T = 6,051$ with $L = 5$, so we can create $6,046$ such (X, Y) pairs.
- We use the first $4,281$ as training data, fitting an RNN with 12 hidden units per lag step (per A_l).

RNN Results for NYSE



$R^2 = 0.42$ for the RNN; $R^2 = 0.18$ for the *straw man*, where yesterday's value of `log trading volume` is used alone to predict today's value.

Autoregression

- The RNN forecaster we just saw is similar in structure to a traditional *autoregression* procedure.

$$y = \begin{bmatrix} v_{L+1} \\ v_{L+2} \\ v_{L+3} \\ \vdots \\ v_T \end{bmatrix} \quad M = \begin{bmatrix} 1 & v_L & v_{L-1} & \cdots & v_1 \\ 1 & v_{L+1} & v_L & \cdots & v_2 \\ 1 & v_{L+2} & v_{L+1} & \cdots & v_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & v_{T-1} & v_{T-1} & \cdots & v_{T-L} \end{bmatrix}$$

Fir an OLS regression of y and M, yielding

$$\hat{v} = \hat{\beta}_0 + \hat{\beta}_1 v_{t-1} + \hat{\beta}_2 v_{t-2} + \cdots + \hat{\beta}_L v_{t-L}$$

This is known as an *order-L autoregression model* or AR(L).

Autoregression on NYSE

We can include lagged versions of `DJ_return` and `log_volatility` in the matrix M, resulting in $3L + 1$ columns

- $R^2 = 0.41$ for AR(5) model (16 parameters)
- $R^2 = 0.42$ for RNN model (205 parameters)
- $R^2 = 0.42$ for AR(5) model fit by neural network
- $R^2 = 0.46$ for all models if we include in the model the day of the week being predicted.

Summary of RNNs

- Many, much more complex, variations of RNNs exist.
- One variation treats the sequence as a one-dimensional image, and uses CNNs for fitting.
  - For example, a sequence of words using an embedded representation can be viewed as an image, and the CNN convolves by sliding a convolutional filter along the sequence.
- Can have additional hidden layers, where each hidden layer is a sequence and treats the previous hidden layer as an input sequence.
- Can have output also be a sequence, and input and output share the hidden units (used for language translation).