LSTM: A Search Space Odyssey(Summary)

1. **Introduction:**
   Recurrent neural networks with Long Short-Term Memory(which we will concisely refer to as LSTMs) have emerged as an effective and scalable model for several learning problems related to sequential data. They do not suffer from the optimization hurdles that plague simple recurrent networks and have been used to advance the state-of-the-art for many difficult problems.

2. **Vanilla LSTM:**
   Detailed Diagram:

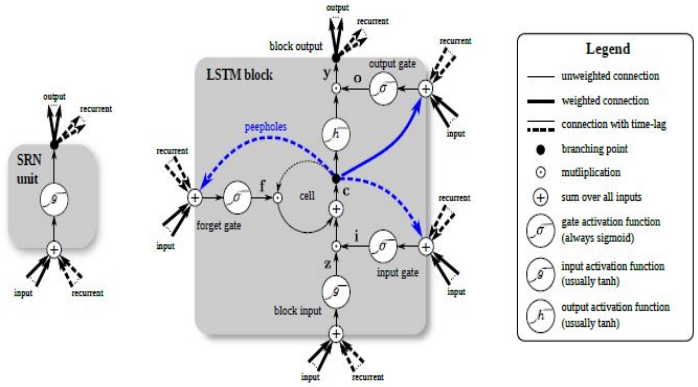

Figure 1. Detailed schematic of the Simple Recurrent Network (SRN) unit (left) and a Long Short-Term Memory block (right) as used in the hidden layers of a recurrent neural network.

Let's look at it mathematically,

A. Forward Pass

Let $\mathbf{x}^t$ be the input vector at time t, N be the number of LSTM blocks and M the number of inputs. Then we get the following weights for an LSTM layer:

- Input weights: $\mathbf{W}_z$, $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o \in \mathbb{R}^{N \times M}$
- Recurrent weights: $\mathbf{R}_z$, $\mathbf{R}_i$, $\mathbf{R}_f$, $\mathbf{R}_o \in \mathbb{R}^{N \times N}$
- Peephole weights: $\mathbf{p}_i$, $\mathbf{p}_f$, $\mathbf{p}_o \in \mathbb{R}^N$
- Bias weights: $\mathbf{b}_z$, $\mathbf{b}_i$, $\mathbf{b}_f$, $\mathbf{b}_o \in \mathbb{R}^N$

Then the vector formulas for a vanilla LSTM layer forward pass can be written as:

$$
\begin{aligned}
\bar{\mathbf{z}}^t &= \mathbf{W}_z \mathbf{x}^t + \mathbf{R}_z \mathbf{y}^{t-1} + \mathbf{b}_z \\
\mathbf{z}^t &= g(\bar{\mathbf{z}}^t) && \textit{block input} \\
\bar{\mathbf{i}}^t &= \mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i \\
\mathbf{i}^t &= \sigma(\bar{\mathbf{i}}^t) && \textit{input gate} \\
\bar{\mathbf{f}}^t &= \mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f \\
\mathbf{f}^t &= \sigma(\bar{\mathbf{f}}^t) && \textit{forget gate} \\
\mathbf{c}^t &= \mathbf{z}^t \odot \mathbf{i}^t + \mathbf{c}^{t-1} \odot \mathbf{f}^t && \textit{cell} \\
\bar{\mathbf{o}}^t &= \mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^t + \mathbf{b}_o \\
\mathbf{o}^t &= \sigma(\bar{\mathbf{o}}^t) && \textit{output gate} \\
\mathbf{y}^t &= h(\mathbf{c}^t) \odot \mathbf{o}^t && \textit{block output}
\end{aligned}
$$

Where σ, g and h are point-wise non-linear activation functions. The *logistic sigmoid* ($\sigma(x) = \frac{1}{1+e^{-x}}$) is used as gate activation function and the *hyperbolic tangent* ($g(x) = h(x) = \tanh(x)$) is usually used as the block input and output activation function. Point-wise multiplication of two vectors is denoted by $\odot$.

B. Backpropagation Through Time

The deltas inside the LSTM block are then calculated as:

$$
\begin{aligned}
\delta \mathbf{y}^t &= \Delta^t + \mathbf{R}_z^T \delta \mathbf{z}^{t+1} + \mathbf{R}_i^T \delta \mathbf{i}^{t+1} + \mathbf{R}_f^T \delta \mathbf{f}^{t+1} + \mathbf{R}_o^T \delta \mathbf{o}^{t+1} \\
\delta \bar{\mathbf{o}}^t &= \delta \mathbf{y}^t \odot h(\mathbf{c}^t) \odot \sigma'(\bar{\mathbf{o}}^t) \\
\delta \mathbf{c}^t &= \delta \mathbf{y}^t \odot \mathbf{o}^t \odot h'(\mathbf{c}^t) + \mathbf{p}_o \odot \delta \bar{\mathbf{o}}^t + \mathbf{p}_i \odot \delta \bar{\mathbf{i}}^{t+1} \\
&\quad + \mathbf{p}_f \odot \delta \bar{\mathbf{f}}^{t+1} + \delta \mathbf{c}^{t+1} \odot \mathbf{f}^{t+1} \\
\delta \bar{\mathbf{f}}^t &= \delta \mathbf{c}^t \odot \mathbf{c}^{t-1} \odot \sigma'(\bar{\mathbf{f}}^t) \\
\delta \bar{\mathbf{i}}^t &= \delta \mathbf{c}^t \odot \mathbf{z}^t \odot \sigma'(\bar{\mathbf{i}}^t) \\
\delta \bar{\mathbf{z}}^t &= \delta \mathbf{c}^t \odot \mathbf{i}^t \odot g'(\bar{\mathbf{z}}^t)
\end{aligned}
$$

Here Δ^t is the vector of deltas passed down from the layer above. If E is the loss function it formally corresponds to $\frac{\partial E}{\partial \mathbf{y}^t}$, but not including the recurrent dependencies. The deltas for the inputs are only needed if there is a layer below that needs training, and can be computed as follows:

$$
\delta \mathbf{x}^t = \mathbf{W}_z^T \delta \bar{\mathbf{z}}^t + \mathbf{W}_i^T \delta \bar{\mathbf{i}}^t + \mathbf{W}_f^T \delta \bar{\mathbf{f}}^t + \mathbf{W}_o^T \delta \bar{\mathbf{o}}^t
$$

Finally, the gradients for the weights are calculated as follows, where $\star$ can be any of $\{\bar{\mathbf{z}}, \bar{\mathbf{i}}, \bar{\mathbf{f}}, \bar{\mathbf{o}}\}$, and $\langle \star_1, \star_2 \rangle$ denotes the outer product of two vectors:

$$
\begin{aligned}
\delta \mathbf{W}_\star &= \sum_{t=0}^{T} \langle \delta \star^t, \mathbf{x}^t \rangle & \delta \mathbf{p}_i &= \sum_{t=0}^{T-1} \mathbf{c}^t \odot \delta \bar{\mathbf{i}}^{t+1} \\
\delta \mathbf{R}_\star &= \sum_{t=0}^{T-1} \langle \delta \star^{t+1}, \mathbf{y}^t \rangle & \delta \mathbf{p}_f &= \sum_{t=0}^{T-1} \mathbf{c}^t \odot \delta \bar{\mathbf{f}}^{t+1} \\
\delta \mathbf{b}_\star &= \sum_{t=0}^{T} \delta \star^t & \delta \mathbf{p}_o &= \sum_{t=0}^{T} \mathbf{c}^t \odot \delta \bar{\mathbf{o}}^t
\end{aligned}
$$

3. Evaluation Step:

Here we are going to compare LSTM variants on 3 different domain datasets with the random search of datasets.

1. Datasets, their network architecture, and training:
    - TIMIT(speech corpus)
        - 61 phones
        - From the raw audio, we extract 12 Mel Frequency Cepstrum Coefficients (MFCCs) [35] + energy over 25ms hamming-windows with a stride of 10ms and a pre-emphasis coefficient of 0.97.
        - 39 normalized input to the network.
        - The training, testing, and validation sets are split into 3696, 400, and 192 sequences, having 304 frames on average.
        - Bidirectional
        - LSTM was used, consisting of two hidden layers, one processing the input forwards and the other one backwards in time.
        - Loss function-cross entropy error.

    - IAM online(handwriting database):
        - The IAM-OnDB dataset splits into one training set, two validation sets, and one test set, having 775, 192, 216, and 544 boards each.

- The networks were trained using the Connectionist Temporal Classification (CTC) error function with 82 outputs (81 characters plus the special empty label).
- Bidirectional LSTM was used for TIMIT and IAM Online tasks, consisting of two hidden layers, one processing the input forwards and the other one backward in time.

- JSB chorals(acoustic modeling):
  - The networks were trained to do next-step prediction by minimizing the negative log-likelihood.
  - The complete dataset consists of 229, 76, and 77 sequences (training, validation, and test sets respectively) with an average length of 61.
  - a single LSTM hidden layer and a sigmoid output layer.
  - Loss function- cross-entropy error.

Notes: The initial weights for all networks were drawn from a normal distribution with a standard deviation of 0.1. The training was done using Stochastic Gradient Descent with Nesterov-style momentum. The learning rate was rescaled by a factor of (1 - momentum). Gradients were computed using full BPTT for LSTMs. Training stopped after 150 epochs or once there was no improvement on the validation set for more than fifteen epochs.

2. LSTM variants to analyze

NIG: No Input Gate: $\mathbf{i}^t = \mathbf{1}$
NFG: No Forget Gate: $\mathbf{f}^t = \mathbf{1}$
NOG: No Output Gate: $\mathbf{o}^t = \mathbf{1}$
NIAF: No Input Activation Function: $g(\mathbf{x}) = \mathbf{x}$
NOAF: No Output Activation Function: $h(\mathbf{x}) = \mathbf{x}$
CIFG: Coupled Input and Forget Gate: $\mathbf{f}^t = \mathbf{1} - \mathbf{i}^t$
NP: No Peepholes:

$$\begin{aligned}\bar{\mathbf{i}}^t &= \mathbf{W}_i\mathbf{x}^t + \mathbf{R}_i\mathbf{y}^{t-1} + \mathbf{b}_i \\ \bar{\mathbf{f}}^t &= \mathbf{W}_f\mathbf{x}^t + \mathbf{R}_f\mathbf{y}^{t-1} + \mathbf{b}_f \\ \bar{\mathbf{o}}^t &= \mathbf{W}_o\mathbf{x}^t + \mathbf{R}_o\mathbf{y}^{t-1} + \mathbf{b}_o\end{aligned}$$

FGR: Full Gate Recurrence:

$$\begin{aligned}\bar{\mathbf{i}}^t &= \mathbf{W}_i\mathbf{x}^t + \mathbf{R}_i\mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i \\ &\quad + \mathbf{R}_{ii}\mathbf{i}^{t-1} + \mathbf{R}_{fi}\mathbf{f}^{t-1} + \mathbf{R}_{oi}\mathbf{o}^{t-1} \\ \bar{\mathbf{f}}^t &= \mathbf{W}_f\mathbf{x}^t + \mathbf{R}_f\mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f \\ &\quad + \mathbf{R}_{if}\mathbf{i}^{t-1} + \mathbf{R}_{ff}\mathbf{f}^{t-1} + \mathbf{R}_{of}\mathbf{o}^{t-1} \\ \bar{\mathbf{o}}^t &= \mathbf{W}_o\mathbf{x}^t + \mathbf{R}_o\mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^{t-1} + \mathbf{b}_o \\ &\quad + \mathbf{R}_{io}\mathbf{i}^{t-1} + \mathbf{R}_{fo}\mathbf{f}^{t-1} + \mathbf{R}_{oo}\mathbf{o}^{t-1}\end{aligned}$$

3. Hyperparameter search
   There we 27 random searches on each variant with base variant thus total number of trials was 9*3(datasets)*200=5400 trials.
   - number of LSTM blocks per hidden layer: log-uniform samples from $[20, 200]$;
   - learning rate: log-uniform samples from $[10^{-6}, 10^{-2}]$;
   - momentum: $1 -$ log-uniform samples from $[0.01, 1.0]$;
   - standard deviation of Gaussian input noise: uniform samples from $[0, 1]$.

4. Results and discussions

- Comparison of variants
    - removing the output activation function (NOAF) or the forget gate (NFG) significantly hurt performance on all three datasets.
    - Input and forget gate coupling (CIFG) did not significantly change mean performance on any of the datasets.
    - Removing peephole connections (NP) also did not lead to significant changes.
    - Adding full gate recurrence (FGR) did not significantly change performance on TIMIT or IAM Online, but led to worse results on the JSB Chorales dataset.

      For supervised learning on continuous real-valued data (such as speech and handwriting recognition), the input gate, output gate, and input activation function are all crucial for obtaining good performance.

- Impact of hyperparameters
  The impact of the learning rate, hidden layer size, input noise is shown in the figure.

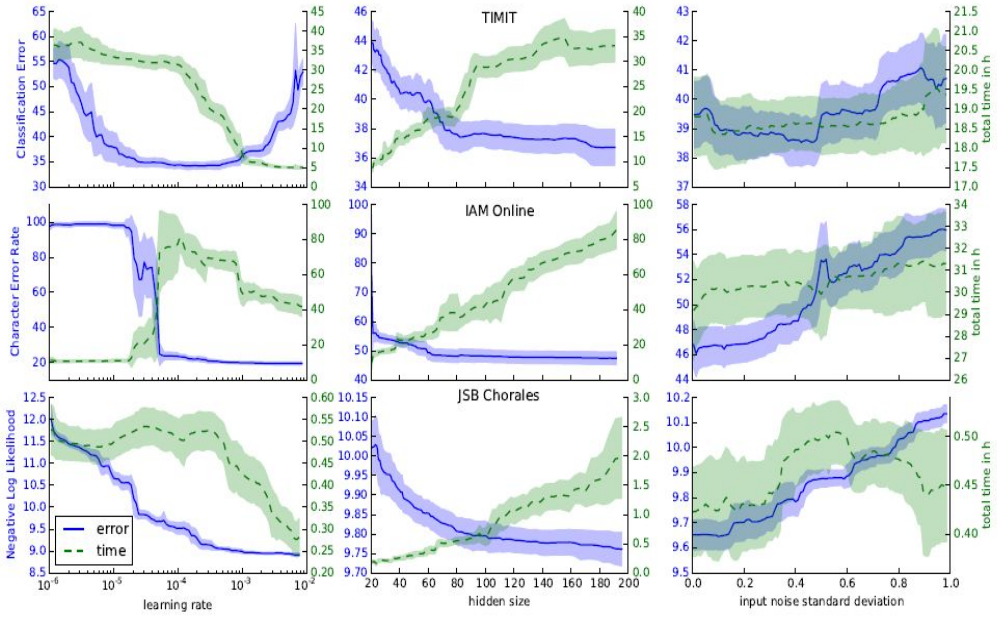

Figure 4. Predicted marginal error (blue) and marginal time for different values of the *learning rate*, *hidden size*, and the *input noise* (columns) for the test set of all three datasets (rows). The shaded area indicates the standard deviation between the tree-predicted marginals and thus the reliability of the predicted mean performance. Note that each plot is for the vanilla LSTM but curves for all variants that are not significantly worse look very similar.

Momentum nor effects performance nor training time in significant way.

5. **Conclusion**
   - Most commonly used LSTM architecture (vanilla LSTM) performs reasonably well on various datasets.
   - Certain modifications such as coupling the input and forget gates (CIFG) or removing peephole connections (NP) simplified LSTMs in our experiments without significantly decreasing performance.
   - The forget gate and the output activation function are the most critical components of the LSTM block.
   - For practical purposes the hyperparameters can be treated as approximately independent.
   - Learning rate is the most crucial hyperparameter, followed by the network size.