Personal Proceeding on Time Series

--DTW, Viz of RNN, Clockwalk RNN Revisiting

(Mar 08, 2017)

YANG Jiancheng

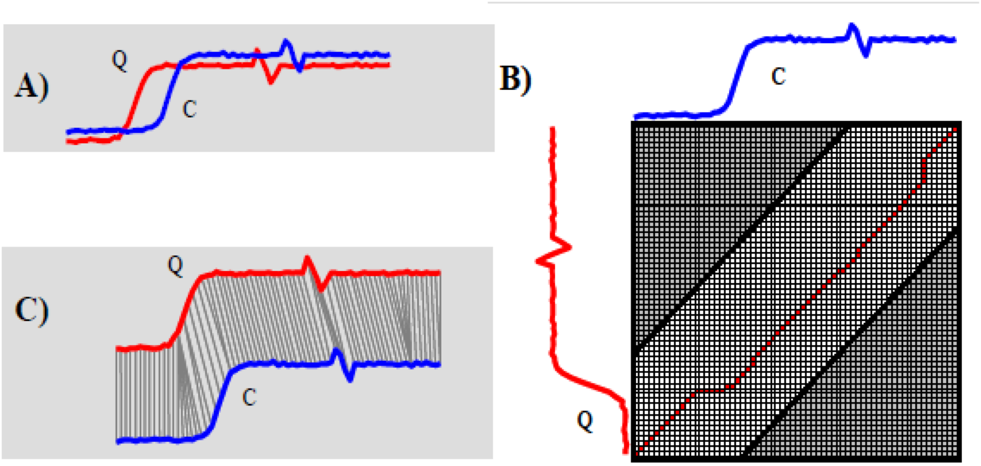
Outline

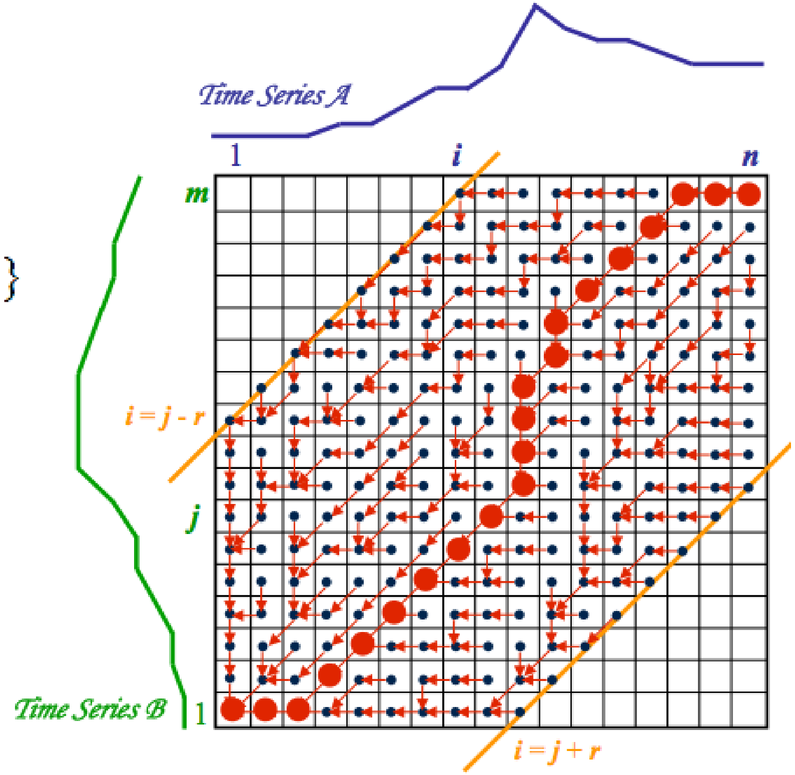
- I. Dynamic Time Wrapping (DTW)
- II. Visualizing and Understanding Recurrent Networks
- III. Revisit Clockwalk RNN
- IV. GEFCom 2012 & 2014

• I. Dynamic Time Wrapping (DTW)

- **Brief Review**

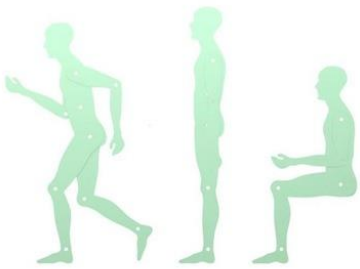
$$\gamma(i,j) = d(q_i,c_j) + \min\{\, \gamma(i\text{-}1,j\text{-}1)\ ,\ \gamma(i\text{-}1,j\,)\ ,\ \gamma(i,j\text{-}1)\,\}$$

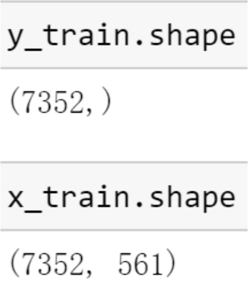




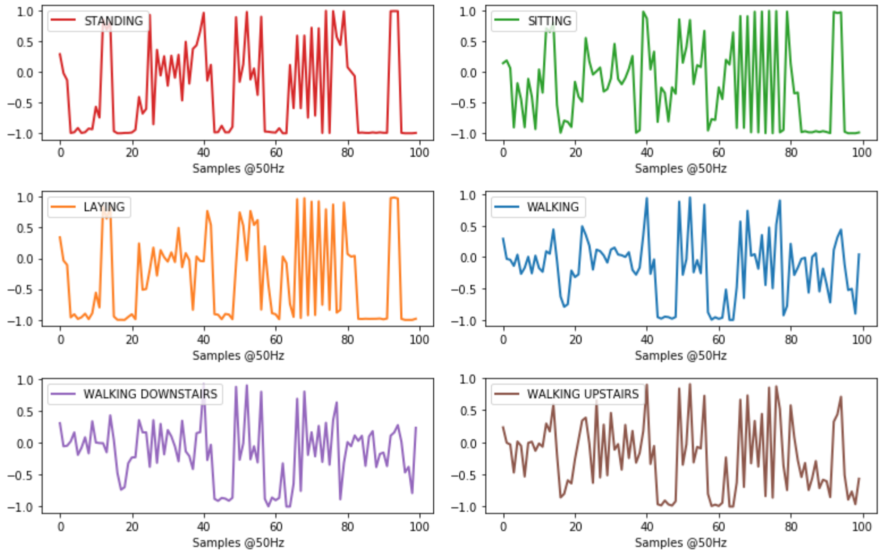
• I. Dynamic Time Wrapping (DTW)

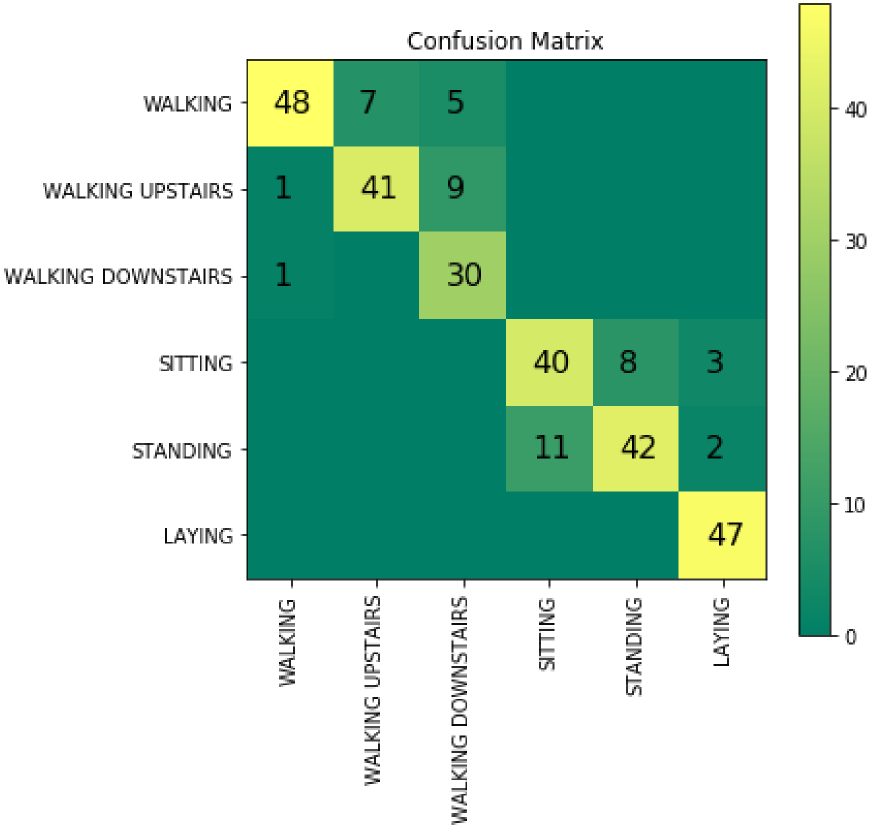
- **Example: Human Activity Recognition Dataset (HAR)**





```
y_train.shape
(7352,)
x_train.shape
(7352, 561)
```

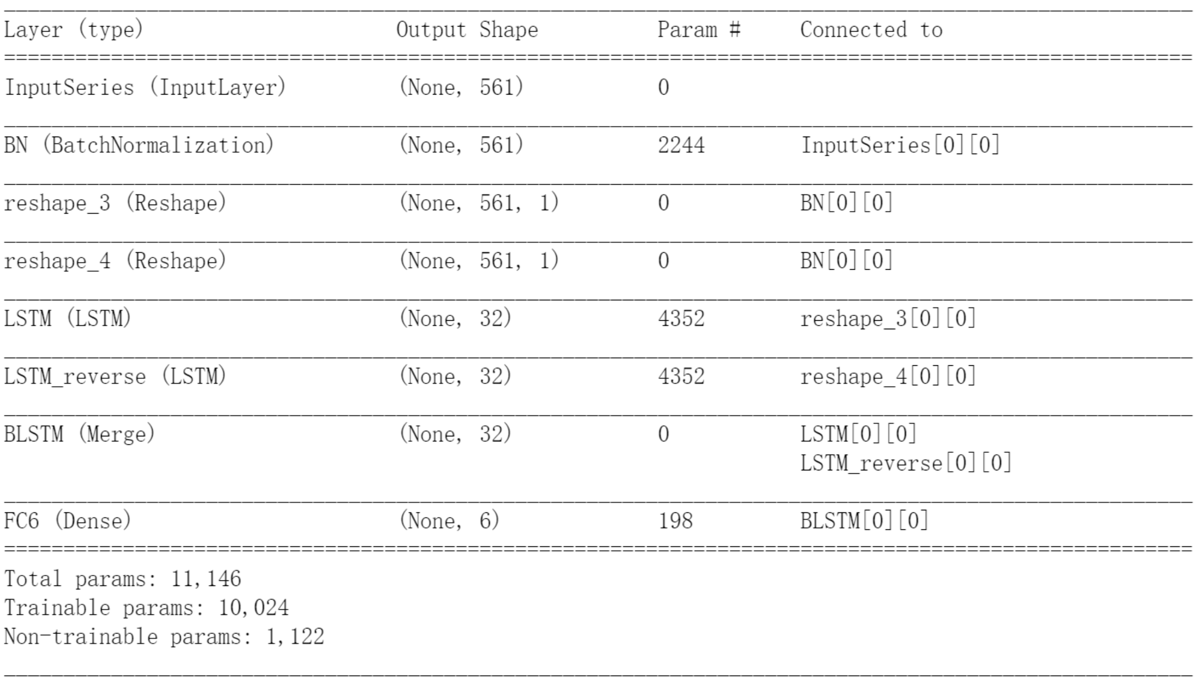




DTW & kNN: ~86%
claimed to be better than the published paper.

• I. Dynamic Time Wrapping (DTW)

- **However…**

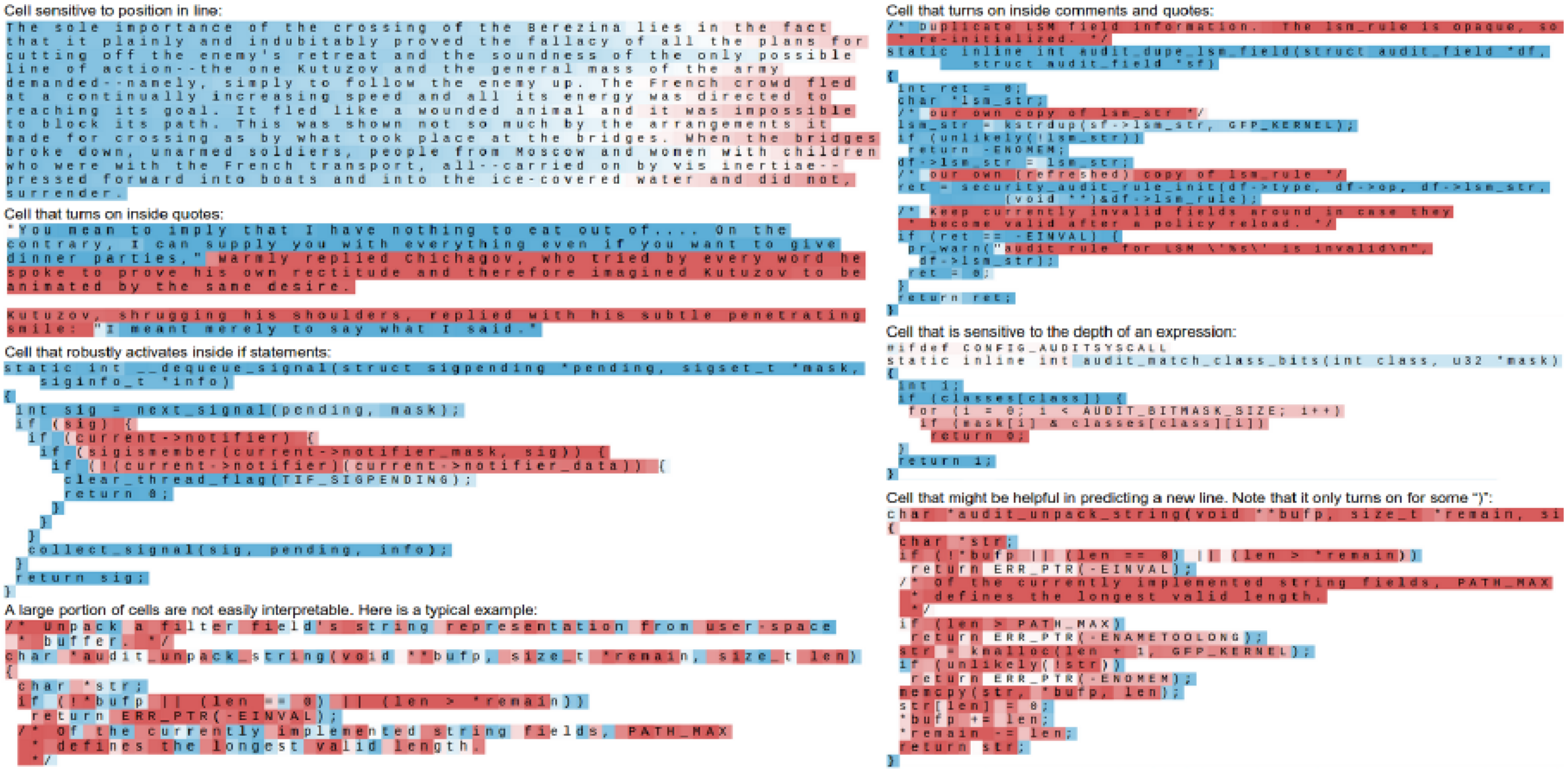


```
____________________________________________________________________________________________________
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
InputSeries (InputLayer)         (None, 561)           0
____________________________________________________________________________________________________
BN (BatchNormalization)          (None, 561)           2244        InputSeries[0][0]
____________________________________________________________________________________________________
reshape_3 (Reshape)              (None, 561, 1)        0           BN[0][0]
____________________________________________________________________________________________________
reshape_4 (Reshape)              (None, 561, 1)        0           BN[0][0]
____________________________________________________________________________________________________
LSTM (LSTM)                      (None, 32)            4352        reshape_3[0][0]
____________________________________________________________________________________________________
LSTM_reverse (LSTM)              (None, 32)            4352        reshape_4[0][0]
____________________________________________________________________________________________________
BLSTM (Merge)                    (None, 32)            0           LSTM[0][0]
                                                                   LSTM_reverse[0][0]
____________________________________________________________________________________________________
FC6 (Dense)                      (None, 6)             198         BLSTM[0][0]
====================================================================================================
Total params: 11,146
Trainable params: 10,024
Non-trainable params: 1,122
____________________________________________________________________________________________________
```

A very simple BLSTM outperforms DTW.
Mine: ~89%
Some optimized work: ~91%

• II. Visualizing and Understanding Recurrent Networks

- **LSTM cells' outputs**



Cell sensitive to position in line:

The sole importance of the crossing of the Berezina lies in the fact
that it plainly and indubitably proved the fallacy of all the plans for
cutting off the enemy's retreat and the soundness of the only possible
line of action--the one Kutuzov and the general mass of the army
demanded--namely, simply to follow the enemy up. The French crowd fled
at a continually increasing speed and all its energy was directed to
reaching its goal. It fled like a wounded animal and it was impossible
to block its path. This was shown not so much by the arrangements it
made for crossing as by what took place at the bridges. When the bridges
broke down, unarmed soldiers, people from Moscow and women with children
who were with the French transport, all--carried on by vis inertiae--
pressed forward into boats and into the ice-covered water and did not,
surrender.

Cell that turns on inside quotes:

"You mean to imply that I have nothing to eat out of.... On the
contrary, I can supply you with everything even if you want to give
dinner parties," warmly replied Chichagov, who tried by every word he
spoke to prove his own rectitude and therefore imagined Kutuzov to be
animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating
smile: "I meant merely to say what I said."

Cell that robustly activates inside if statements:

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
    siginfo_t *info)
{
  int sig = next_signal(pending, mask);
  if (sig) {
    if (current->notifier) {
      if (sigismember(current->notifier_mask, sig)) {
        if (!(current->notifier)(current->notifier_data)) {
          clear_thread_flag(TIF_SIGPENDING);
          return 0;
        }
      }
    }
    collect_signal(sig, pending, info);
  }
  return sig;
}
```

A large portion of cells are not easily interpretable. Here is a typical example:

```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
   * defines the longest valid length.
   */
```

Cell that turns on inside comments and quotes:

```
/* Duplicate LSM field information.  The lsm_rule is opaque, so
 * re-initialized. */
static inline int audit_dupe_lsm_field(struct audit_field *df,
           struct audit_field *sf)
{
  int ret = 0;
  char *lsm_str;
  /* our own copy of lsm_str */
  lsm_str = kstrdup(sf->lsm_str, GFP_KERNEL);
  if (unlikely(!lsm_str))
    return -ENOMEM;
  df->lsm_str = lsm_str;
  /* our own (refreshed) copy of lsm_rule */
  ret = security_audit_rule_init(df->type, df->op, df->lsm_str,
             (void **)&df->lsm_rule);
  /* Keep currently invalid fields around in case they
   * become valid after a policy reload. */
  if (ret == -EINVAL) {
    pr_warn("audit rule for LSM \'%s\' is invalid\n",
      df->lsm_str);
    ret = 0;
  }
  return ret;
}
```

Cell that is sensitive to the depth of an expression:

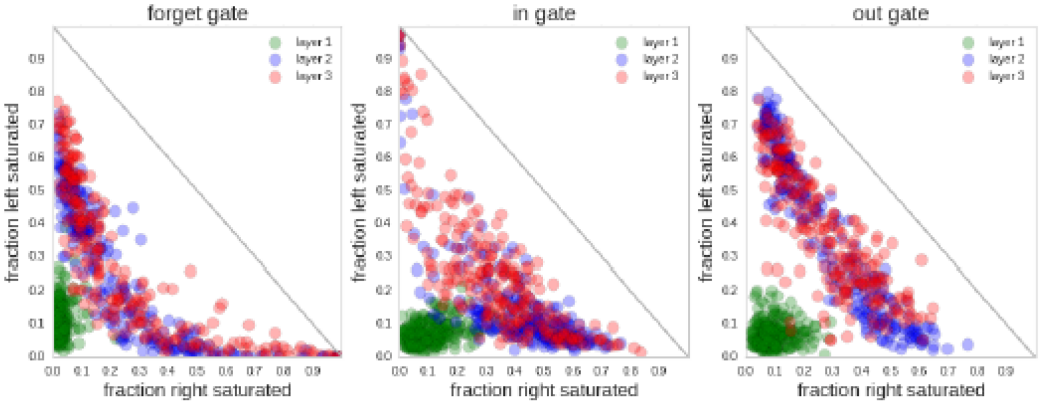
```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
  int i;
  if (classes[class]) {
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
      if (mask[i] & classes[class][i])
        return 0;
  }
  return 1;
}
```

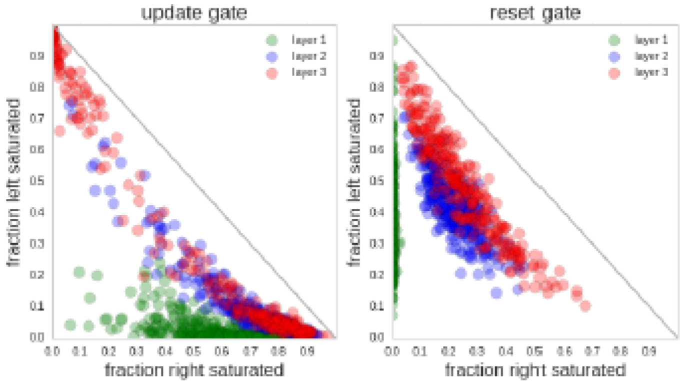
Cell that might be helpful in predicting a new line. Note that it only turns on for some ")":

```
char *audit_unpack_string(void **bufp, size_t *remain, si
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
   * defines the longest valid length.
   */
  if (len > PATH_MAX)
    return ERR_PTR(-ENAMETOOLONG);
  str = kmalloc(len + 1, GFP_KERNEL);
  if (unlikely(!str))
    return ERR_PTR(-ENOMEM);
  memcpy(str, *bufp, len);
  str[len] = 0;
  *bufp += len;
  *remain -= len;
  return str;
}
```

• II. Visualizing and Understanding Recurrent Networks

- **LSTM gates' activation**



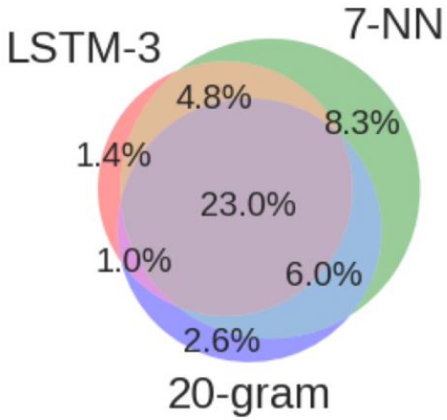


• II. Visualizing and Understanding Recurrent Networks

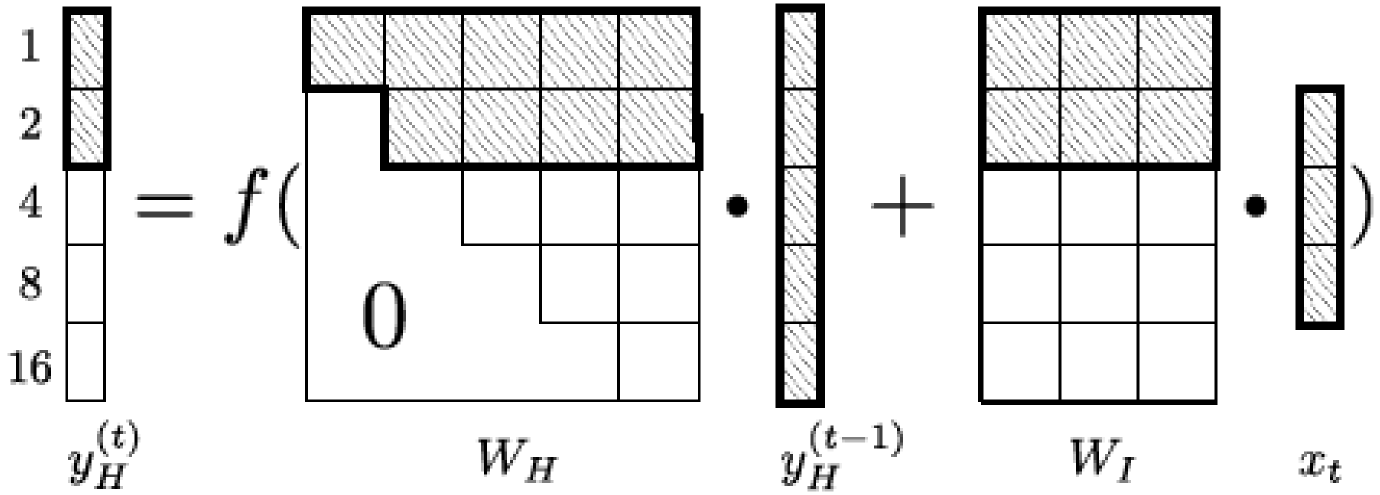
- **Other Highlights**

a) Long-range interaction
LSTM **outperforms in accuracy, cross-entropy and model size**.

b) Break-down failure cases
too specific to NLP, but note that they do this by selecting the error cases manually.
They found simply scaling up the model can reduce the local (n-gram) error, but leave other error untouched.



• III. Revisit Clockwalk RNN



$$\mathbf{y}_H^{(t)} = f_H(\mathbf{W}_H \cdot \mathbf{y}^{(t-1)} + \mathbf{W}_I \cdot \mathbf{x}^{(t)}), \qquad (1)$$

$$\mathbf{y}_O^{(t)} = f_O(\mathbf{W}_O \cdot \mathbf{y}_H^{(t)}), \qquad (2)$$

$$\mathbf{W}_H = \begin{pmatrix} \mathbf{W}_{H_1} \\ \vdots \\ \mathbf{W}_{H_g} \end{pmatrix} \qquad \mathbf{W}_I = \begin{pmatrix} \mathbf{W}_{I_1} \\ \vdots \\ \mathbf{W}_{I_g} \end{pmatrix} \qquad (3)$$

• IV. GEFCom 2012 & 2014

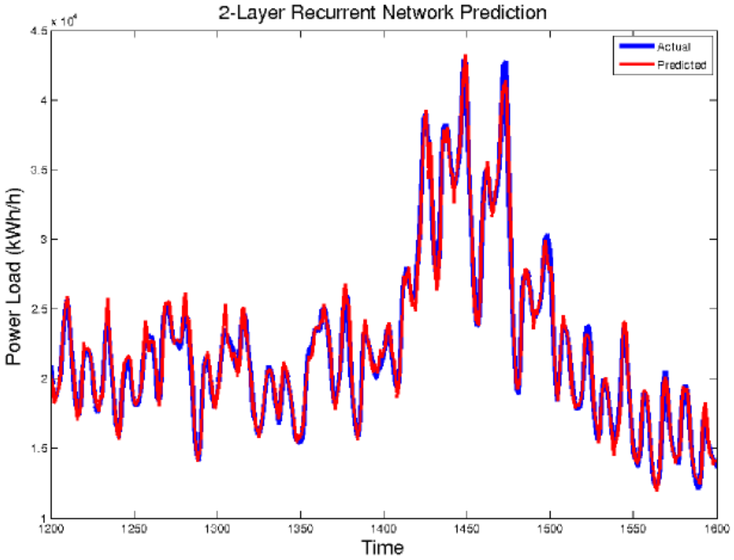
a) Load Predicting

b) Wind Predicting

c) Price Predicting

d) Solar Predicting

e) 2014: Probabilistic Forecasting



Bibliography

- A Clockwork RNN (arXiv)
- Visualizing and Understanding Recurrent Networks (arXiv)
- K Nearest Neighbors & Dynamic Time Warping (code)
- Everything you know about Dynamic Time Warping is Wrong (link)

Thanks for listening!