On Machine Learning & Analyzing and Predicting Software Bug's with a Hidden Markov Model

Thomas Stibor
t.stibor@gsi.de

High Performance Computing
GSI Helmholtz Centre for Heavy Ion Research
Darmstadt, Germany

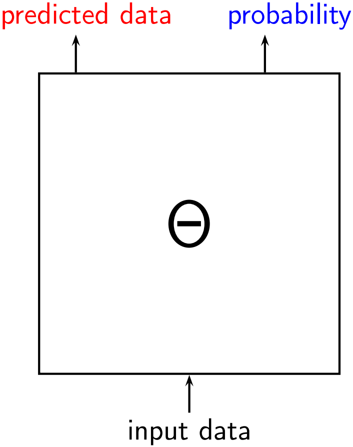
Thursday 28th June, 2018

Data Science and Quantum Computing Workshop
TRIUMF, Vancouver, Canada

Machine Learning Framework

Machine Learning $\equiv$ Optimization & Statistics

Data $\equiv$ (input data, target data)



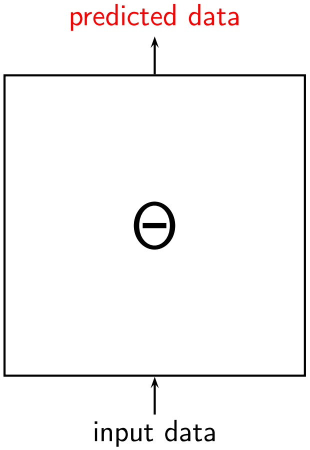

```
while not min Loss_Θ(target data, predicted data) {
    fit parameters Θ
}

while not max Prob(target data, input data | Θ) {
    fit parameters Θ
}
```

Machine Learning Framework (Ex. Neural Networks (NN))

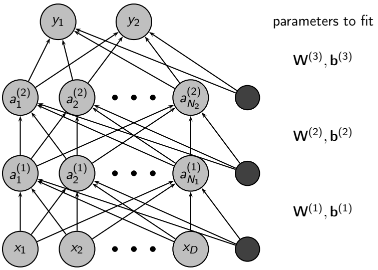
Machine Learning $\equiv$ Optimization & Statistics

Data $\equiv$ (input data $\mathbf{X}$, target data $\mathbf{Y}$)



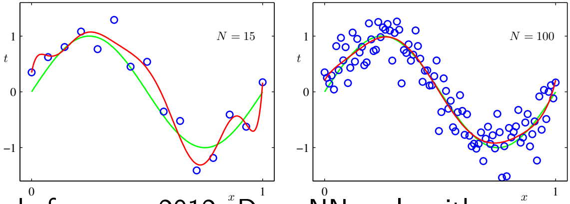

```
while not min Loss_Θ(input data, predicted data) {
    fit parameters Θ := W^(1,2,3), b^(1,2,3) (matrices, vectors)
}
```

minimize $\frac{1}{2}\|f(\mathbf{W}^{(3)}f(\mathbf{W}^{(2)}f(\mathbf{W}^{(1)}\mathbf{X}+\mathbf{b}^{(1)})+\mathbf{b}^{(2)})+\mathbf{b}^{(3)})-\mathbf{Y}\|^2$

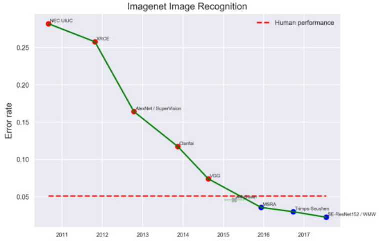


Machine Learning (Scale and Regularize with Data)

Known before: More data helps prevent overfitting (regularize).
Example 9^{th} order polynomial fitting with N data points.



Not known before year 2012: Deep NN scale with amount of data.



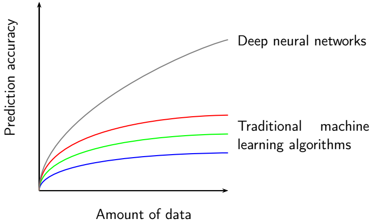
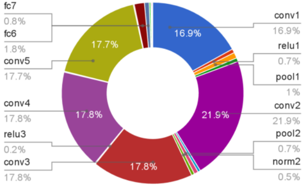


Image from: www.eff.org/ai/metrics

Large-scale Data and Matrix Multiplication

Pete Warden: I spend most of my time worrying about how to make deep learning with neural networks faster and more power efficient. In practice that means focusing on a function called GEMM (GEneral Matrix to Matrix Multiplication).



Taken from: UCB/EECS-2014-93

Computation time distribution of individual layer for the forward pass. Almost all the time (95%) are spent in fc/conv layers implemented using GEMM.

Basically *all* deployed machine learning algorithms are "vectorized" and employ matrix/scalar multiplications and we scale currently just with faster hardware.

Large-scale Data and Matrix Multiplication (cont.)

```
void mult_naive(double A[dim][dim], double B[dim][dim], double C[dim][dim],
     unsigned int dim)
{
    for (unsigned int i = 0; i < dim; i++) {
        for (unsigned int j = 0; j < dim; j++) {
            double sum = 0;
            for (unsigned int k = 0; k < dim; k++) {
                sum += A[i][k] * B[k][j];
            }
            C[i][j] = sum;
        }
    }
}
```

Run-time complexities big $\mathcal{O}$ notation for matrix multiplication algorithms:

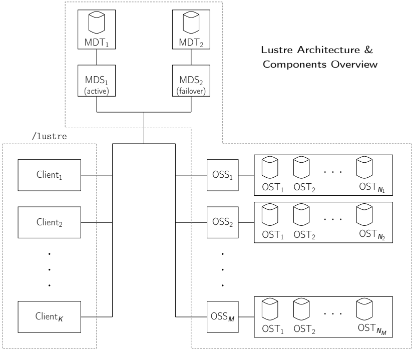
$$\begin{aligned} \text{Naive example} &\quad : \quad \mathcal{O}(dim^3) \\ \text{Strassen(1969)} &\quad : \quad \mathcal{O}(dim^{2.8074}) \\ \text{Coppersmith–Winograd(1990)} &\quad : \quad \mathcal{O}(dim^{2.375477}) \\ \text{Francois Le Gall(2014)} &\quad : \quad \mathcal{O}(dim^{2.3728639}) \end{aligned}$$

If machine learning performance (prediction accuracy) is scaled with data, then new computing paradigms are required to feed the next revolution in machine learning.

Analyzing and Predicting Software Bug's with a Hidden Markov Model

Motivation: Can we apply machine learning techniques to model the behavior of software (a parallel and distributed file system) to understand bug occurrences.

- Parallel, distributed object based file system.
- Petabytes of storage capacity.
- Tens of thousands of nodes.
- Hundreds of Gigabytes / second of throughput.



LBUG's and Implications

- Lustre is a large project with complex code base.
- Lustre is prone to critical software bugs called (LBUG's).
- LBUG is a software behavior that causes freeze of kernel thread and subsequent reboot.

```
void lbug_with_loc(struct libcfs_debug_msg_data *) __attribute__((noreturn));

#define LBUG()                                                          \
do {                                                                    \
        LIBCFS_DEBUG_MSG_DATA_DECL(msgdata, D_EMERG, NULL);             \
        lbug_with_loc(&msgdata);                                        \
} while(0)

#define LIBCFS_DEBUG_MSG_DATA_DECL(dataname, mask, cdls)    \
        static struct libcfs_debug_msg_data dataname = {    \
               .msg_subsys = DEBUG_SUBSYSTEM,               \
               .msg_file   = __FILE__,                      \
               .msg_fn     = __FUNCTION__,                  \
               .msg_line   = __LINE__,                      \
               .msg_cdls   = (cdls)         };              \
        dataname.msg_mask   = (mask);
```

"Note - LBUG freezes the thread to allow capture of the panic stack. A system reboot is needed to clear the thread."

From: http://wiki.lustre.org/manual/LustreManual20_HTML/LustreTroubleshooting.html

LBUG's in Lustre Code Example

```
int mdt_getxattr(struct mdt_thread_info *info)
{
        struct ptlrpc_request  *req = mdt_info_req(info);
        struct mdt_export_data *med = mdt_req2med(req);
        struct lu_ucred        *uc  = lu_ucred(info->mti_env);
        ...
        ...
        valid = info->mti_body->valid & (OBD_MD_FLXATTR | OBD_MD_FLXATTRLS);

        if (valid == OBD_MD_FLXATTR) {
                char *xattr_name = req_capsule_client_get(info->mti_pill,
                                                           &RMF_NAME);
                rc = mdt_getxattr_one(info, xattr_name, next, buf, med, uc);
        } else if (valid == OBD_MD_FLXATTRLS) {
                CDEBUG(D_INODE, "listxattr\n");

                rc = mo_xattr_list(info->mti_env, next, buf);
                if (rc < 0)
                        CDEBUG(D_INFO, "listxattr failed: %d\n", rc);
        } else if (valid == OBD_MD_FLXATTRALL) {
                rc = mdt_getxattr_all(info, reqbody, repbody,
                                      buf, next);
        } else
                LBUG();
```

Processed Lustre Logs

Total amount of Lustre log data is $\approx$ 2.1 GByte.

Example 1:

```
Mar 26 20:05:28,lxfs290,LustreError,filter.c,2732,__filter_oa2dentry,testlust-
     OST001f: filter_preprw_read on non-existent object: 10
Mar 26 20:05:28,lxfs290,LustreError,filter_io.c,488,filter_preprw_read,ASSERTION
     (PageLocked(lnb->page)) failed
Mar 26 20:05:28,lxfs290,LustreError,filter_io.c,488,filter_preprw_read,LBUG
```

Example 2:

```
May 27 22:56:01,lxfs124,LustreError,events.c,381,server_bulk_callback,"event
     type 4, status -5, desc ffff8800c791c000"
May 28 00:15:50,lxmds11,LustreError,client.c,178,ptlrpc_free_bulk,ASSERTION(
     atomic_read(&(desc->bd_export)->exp_refcount) < 0x5a5a5a) failed
May 28 00:15:50,lxmds11,LustreError,service.c,1426,ptlrpc_server_handle_request,
     ASSERTION(atomic_read(&(export)->exp_refcount) < 0x5a5a5a) failed
May 28 00:15:50,lxmds11,LustreError,service.c,1426,ptlrpc_server_handle_request,
     LBUG
May 28 00:15:50,lxmds11,LustreError,client.c,178,ptlrpc_free_bulk,LBUG
```

Note: Patch exists for Example 2

Lustre / LU-919

Multiple wrong LBUGs checking cfs_atomic_t vars/fields with inacurate poison value of 0x5a5a5a

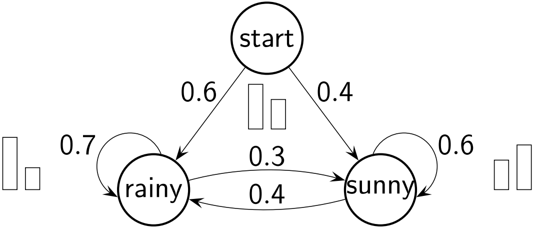
Can we predict LBUG's in Lustre?

Consider this simple approach for modeling Lustre *function calls* and corresponding LBUG occurrence:

$$\begin{tikzcd}
\texttt{mdt_open} \arrow[r] \arrow[d] & \texttt{mdt_fseek} \arrow[r] \arrow[d] & \texttt{mdt_fstat} \arrow[r] \arrow[d] & \texttt{mdt_ost_fid} \arrow[d] \\
\texttt{LBUG}=\text{NO} & \texttt{LBUG}=\text{NO} & \texttt{LBUG}=\text{NO} & \texttt{LBUG}=\text{YES}
\end{tikzcd}$$

This looks like the familiarized Hidden Markov model.

Markov Model Weather Example



$$\mathbf{s} = \begin{array}{cc} \text{rainy} & \text{sunny} \\ (\,0.6\,, & 0.4\,) \end{array}$$

$$\mathbf{H} = \begin{array}{c|cc} & \text{rainy} & \text{sunny} \\ \text{rainy} & 0.7 & 0.3 \\ \text{sunny} & 0.4 & 0.6 \end{array} \quad \text{i.e.} \quad \begin{pmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{pmatrix}$$

T	:	Length of observation sequence.
N_Q	:	Number of states in the model.
$\mathbf{s}$	:	Initial state distribution.
$\mathbf{H}$	:	Transition matrix.
$\{Q_1, Q_2, \ldots, Q_T\}$	:	Set of time indexed random variables for states.
$\mathfrak{M} = (\mathbf{s}, \mathbf{H})$	:	Markov model parameters.

Joint distribution for *sequence* of T observations
$\Pr(Q_1, Q_2, \ldots, Q_T) = \Pr(Q_1) \prod_{t=2}^{T} \Pr(Q_t | Q_1, \ldots, Q_{t-1})$

Joint distribution under *first-order* Markov assumption:
$\Pr(Q_1, Q_2, \ldots, Q_T) = \Pr(Q_1) \prod_{t=2}^{T} \Pr(Q_t | Q_{t-1})$

Markov Model Weather Example (cont.)

Given that weather on day 1 ($t = 1$) is sunny.

- What is the probability for the next 3 days weather will be $\mathcal{O} =$"sunny$\rightarrow$rainy$\rightarrow$rainy"?

$$
\begin{aligned}
\Pr(\mathcal{O}|\mathfrak{M}) &= \Pr(Q_1 = \text{sunny}, Q_2 = \text{sunny}, Q_3 = \text{rainy}, Q_4 = \text{rainy}) \\
&= \Pr(Q_1 = \text{sunny}) \cdot \Pr(Q_2 = \text{sunny} \,|\, Q_1 = \text{sunny}) \\
&\quad \cdot \Pr(Q_3 = \text{rainy} \,|\, Q_2 = \text{sunny}) \\
&\quad \cdot \Pr(Q_4 = \text{rainy} \,|\, Q_3 = \text{rainy}) \\
&= \mathbf{s}_2 \cdot \mathbf{H}_{22} \cdot \mathbf{H}_{21} \cdot \mathbf{H}_{11} \\
&= 0.4 \cdot 0.6 \cdot 0.4 \cdot 0.7 = 0.0672
\end{aligned}
$$

Note: Entries in matrix $\mathbf{H}$ can be interpreted as follows:

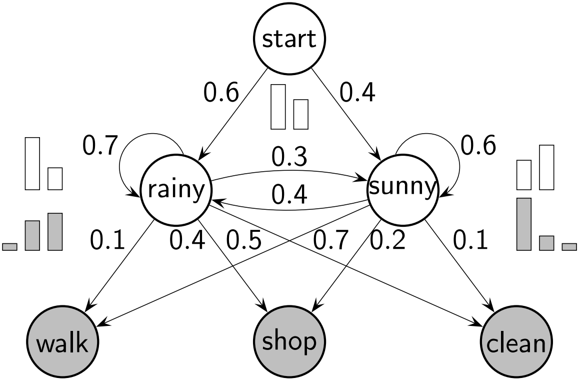
$$\mathbf{H}_{ij} = \Pr(Q_{t+1} = j \,|\, Q_t = i)$$

where for the sake of simplicity states are from set $\{1, 2, \ldots, N_Q\}$

Hidden Markov models

Suppose we are locked in room without windows, and somebody is telling us the following observations and ask us to tell him what weather is outside:

$$\mathcal{O} = \text{walk} \rightarrow \text{clean} \rightarrow \text{shop} \rightarrow \cdots \rightarrow \text{shop}$$



$$\mathbf{s} = \begin{array}{c} \begin{array}{cc} \text{rainy} & \text{sunny} \end{array} \\ \left(\begin{array}{cc} 0.6\ , & 0.4 \end{array} \right) \end{array}$$

$$\mathbf{H} = \begin{array}{c} \\ \text{rainy} \\ \text{sunny} \end{array} \begin{array}{c} \begin{array}{cc} \text{rainy} & \text{sunny} \end{array} \\ \left(\begin{array}{cc} 0.7 & 0.3 \\ 0.4 & 0.6 \end{array} \right) \end{array}$$

$$\mathbf{E} = \begin{array}{c} \\ \text{rainy} \\ \text{sunny} \end{array} \begin{array}{c} \begin{array}{ccc} \text{walk} & \text{shop} & \text{clean} \end{array} \\ \left(\begin{array}{ccc} 0.1 & 0.4 & 0.5 \\ 0.7 & 0.2 & 0.1 \end{array} \right) \end{array}$$

Hidden Markov models (cont.)

Problem 1: Calculate probability of observation sequence $\mathcal{O}$, given model $\mathfrak{M}$, that is $\Pr(\mathcal{O} \mid \mathfrak{M}) = ?$

Problem 2: Given HMM and observation sequence $\mathcal{O}$, find most likely hidden state sequence.

Problem 3: How do we estimate model parameters $\mathfrak{M} = (\mathbf{s}, \mathbf{H}, \mathbf{E})$ to maximize $\Pr(\mathcal{O} \mid \mathfrak{M})$. Loosely speaking, how do we estimate $\mathfrak{M}$ that "best" fits our data $\mathcal{O}$.

For further details see paper:

- *Lawrence R. Rabiner*: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, 1989, pages 257-286.

Analyze Novel Flatland with a HMM

Example from Sec. 1 *Of the Nature of Flatland:* by Edwin A. Abbott

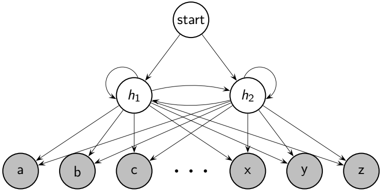
```
I call our world Flatland , not because we call it so , but to make its
nature clearer to you , my happy readers , who are privileged to live in
Space .

Imagine a vast sheet of paper on which straight Lines , Triangles ,
Squares , Pentagons , Hexagons , and other figures , ...
```

Process and convert text into:

```
i c a l l o u r w o r l d f l a t l a n d n o t b e c a u s e w e c a l l i t
s o b u t t o m a k e i t s n a t u r e c l e a r e r t o y o u m y h a p p y
r e a d e r s w h o a r e p r i v i l e g e d t o l i v e i n s p a c e i m a
g i n e a v a s t s h e e t o f p a p e r o n w h i c h s t r a i g h t l i n
e s t r i a n g l e s s q u a r e s p e n t a g o n s h e x a g o n s a n d o
t h e r f i g u r e s
```

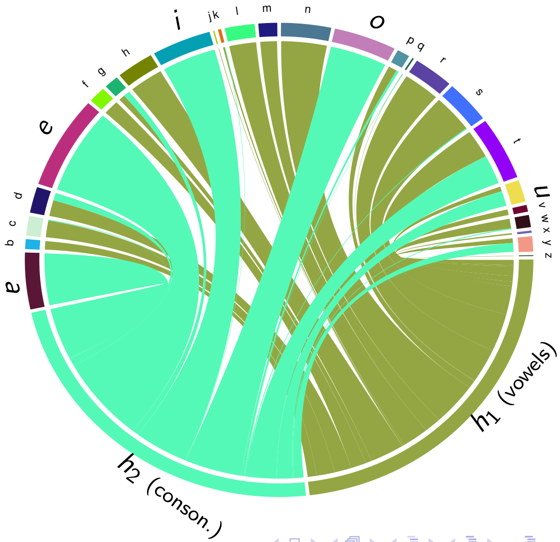
Train HMM of following form:



Analyze Novel Flatland with a HMM (cont.)

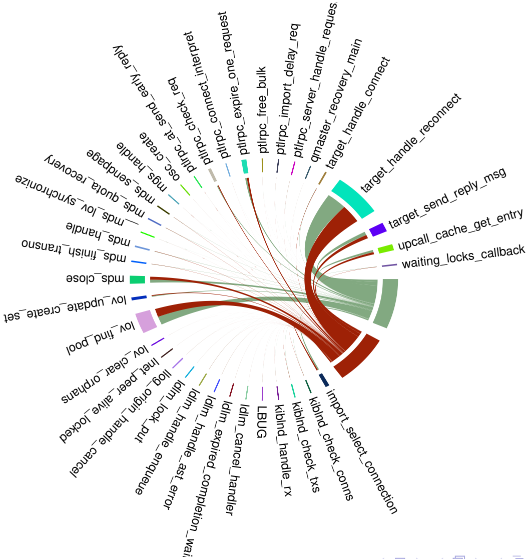
letter	h_1	h_2
a	0.00	16.51
b	2.70	0.00
c	5.37	0.24
d	5.06	2.39
e	0.00	27.10
f	4.59	0.00
g	1.84	1.92
h	10.10	0.00
i	0.00	16.14
j	0.16	0.00
k	0.50	0.16
l	8.02	0.00
m	4.97	0.00
n	13.58	0.00
o	0.00	17.03
p	2.55	0.77
q	0.13	0.13
r	10.90	0.00
s	12.06	0.30
t	9.03	9.29
u	1.49	4.80
v	1.78	0.00
w	3.04	0.34
x	0.46	0.00
y	1.46	2.82
z	0.11	0.00

$\times 100$

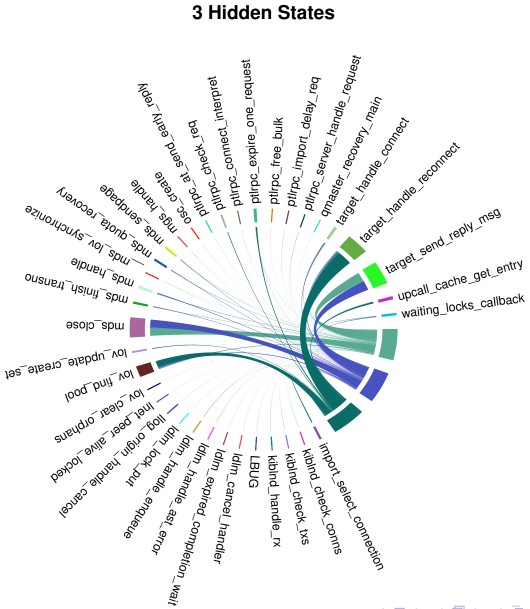


Visualize HMM of Functions Calls

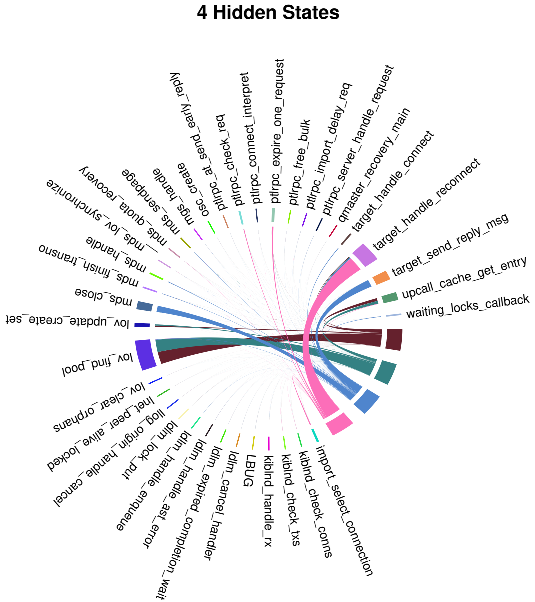
2 Hidden States



Visualize HMM of Functions Calls (cont.)



Visualize HMM of Functions Calls (cont.)

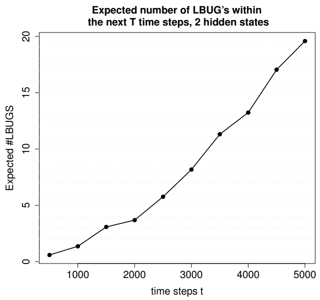


LBUG Prediction with HMM

- Train HMM on Lustre logs (function calls, LBUGs)
- Sample function call sequences from HMM.

Example: Sample from trained HMM (2 hidden states, 36 emitting states (function calls))

```
ldlm_handle_enqueue , lov_find_pool , target_handle_connect ,
target_handle_reconnect , target_handle_reconnect , target_handle_reconnect ,
lov_find_pool , target_send_reply_msg , mds_close ,
lov_clear_orphans , ...., target_handle_reconnect , target_handle_reconnect ,
target_handle_reconnect , LBUG
```



Summary & Outlook

Machine learning framework:

- Optimization & Statistics.
- Scaling with data requires a computing paradigm shift.

Analyzing Lustre log files with (Hidden) Markov Models for:

- visualizing and analyzing problems,
- recover latent structures and relations,
- predicting future problems (`LBUG`'s), by sampling from the model.

HMM implementation (in `C`) and `R` scripts (for generating chord diagrams) is available at https://bitbucket.org/tstibor

Questions ?