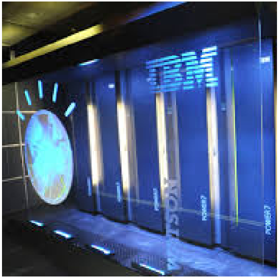
Natural Language Processing

NLPL
Nordic Language
Processing Laboratory

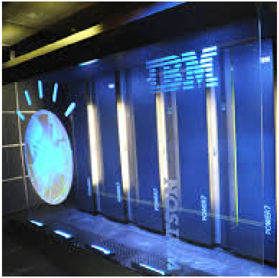
Stephan Oepen
University of Oslo
oe@ifi.uio.no

What is Natural Language Processing (NLP)?

What is Natural Language Processing (NLP)?



What is Natural Language Processing (NLP)?



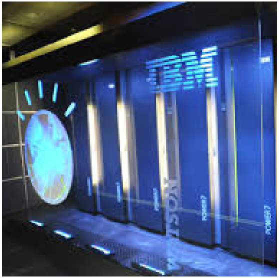
9:41 AM

On May 19 remind me it's dad's birthday ”

Here's your reminder for May 19, 2012 at 9 am:

19 Saturday May 2012

What is Natural Language Processing (NLP)?



On May 19 remind me it's dad's birthday

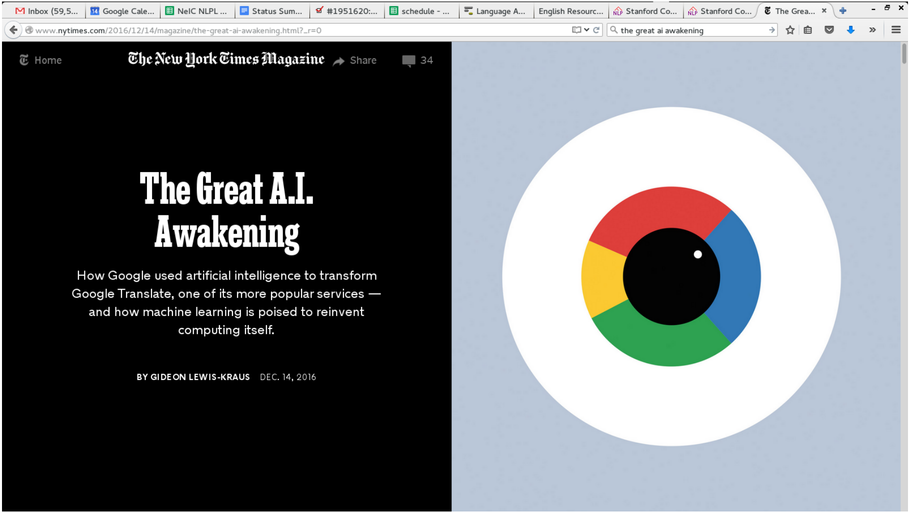
Here's your reminder for 19, 2012 at 9 am:

19 Saturday May 2012

In the Big Picture

- Sub-discipline of CS: Systems that ‘make sense’ of human language;
- core part of (Big) Data Science: Language is the fabric of the Web;
- since 1990s (or so), driven by machine learning: Data and computing.

NLP in the News (Two Years Ago)



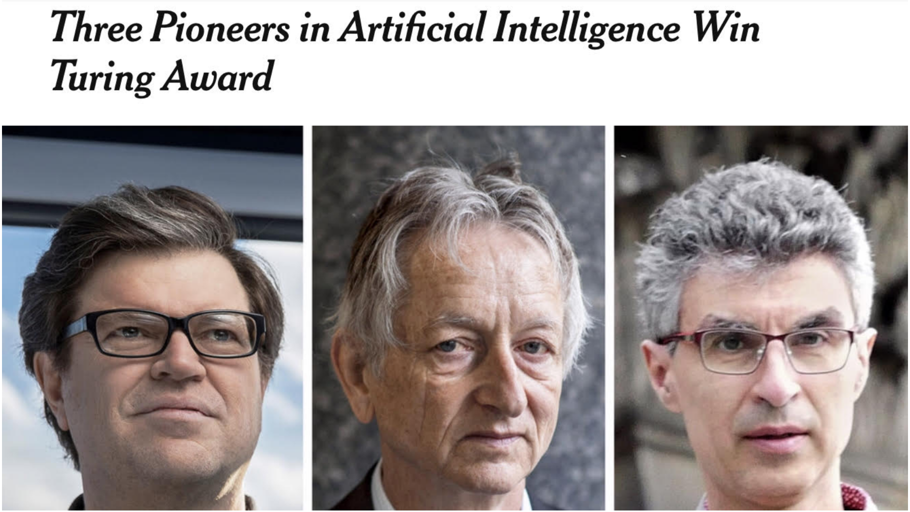
The Great A.I. Awakening

How Google used artificial intelligence to transform Google Translate, one of its more popular services — and how machine learning is poised to reinvent computing itself.

BY GIDEON LEWIS-KRAUS DEC. 14, 2016

(New York Times Magazine, December 2016)

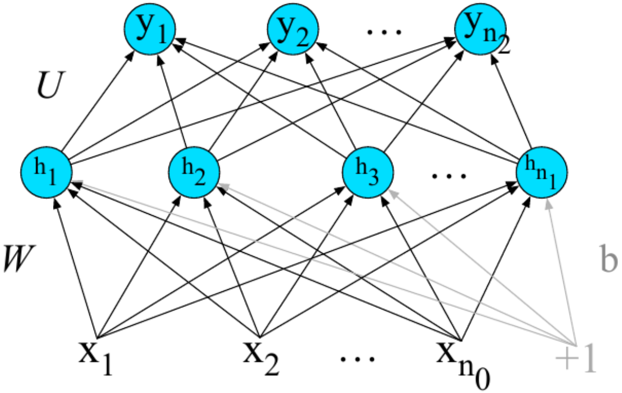
NLP in the News (Last Month)



Three Pioneers in Artificial Intelligence Win Turing Award

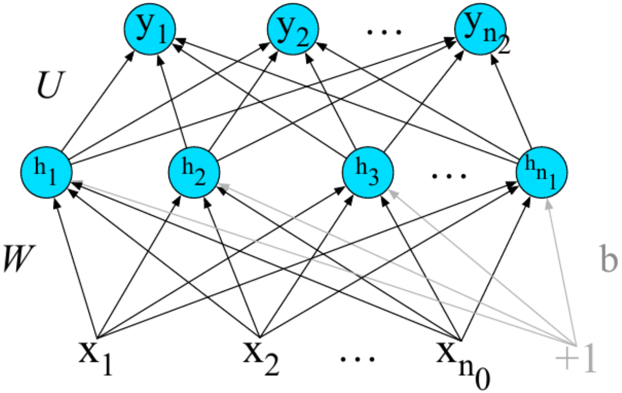
(The New York Times, March 27, 2019)

Machine Learning (Just Now) is ‘Deep’ Learning



- Artificial neural networks: millions of units → large-scale linear algebra.

Machine Learning (Just Now) is ‘Deep’ Learning



- Artificial neural networks: millions of units → large-scale linear algebra.

Where we Come From

Self-help network of NLP developers in Northern Europe;
six university research groups (Denmark, Finland, Sweden, Norway);
national e-infrastructure providers in Finland and Norway.

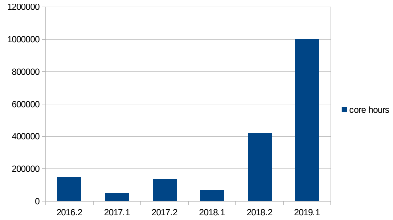
Where we Come From

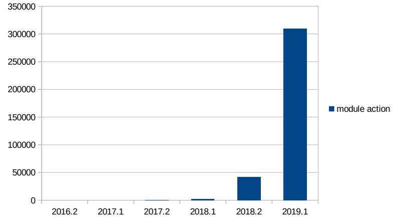
Self-help network of NLP developers in Northern Europe;
six university research groups (Denmark, Finland, Sweden, Norway);
national e-infrastructure providers in Finland and Norway.
Shared allocations on Abel and Taito; discipline-specific software & data;
funded by NeIC 2017–19, matching in-kind contributions by all partners.

NLPL Timeline

May 2012	First Letter of Interest Submitted to NeIC;
May 2013	Invited Presentation at First NeIC Conference;
May 2014	Second Letter of Interest Submitted to NeIC;
November 2014	NLP in the Nordics Workshop (with Common Crawl);
January 2015	Presentation to NeIC All-Hands Meeting;
January 2015	Discussion with NeIC Providers Forum;
2015–16	Project Directive and Collaboration Agreement;
2017–19	NeIC Project: Three Person Years, 50% Co-Funding;
2019–21	Community Use Case in EOSC-Nordic Project.

Community Uptake To Date (on Abel)

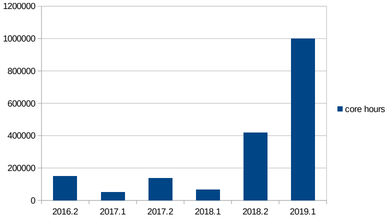


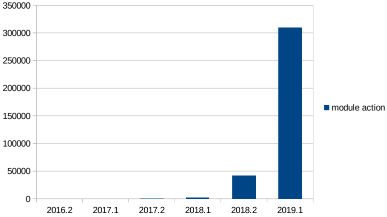


NLPL Core Architecture

- Jointly maintained project directory:
- discipline-specific software and data;
- largely parallel across Abel and Taito;
- access for NLPL users and associates;
- from 'power users' to MSc students;

Community Uptake To Date (on Abel)

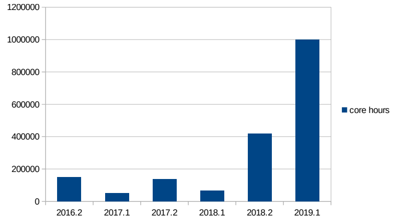


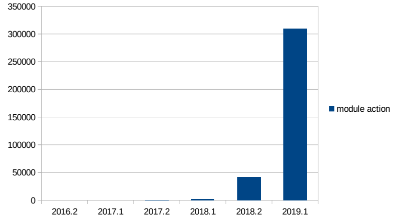


NLPL Core Architecture

- Jointly maintained project directory:
- discipline-specific software and data;
- largely parallel across Abel and Taito;
- access for NLPL users and associates;
- from 'power users' to MSc students;
- maybe 3–4 million core hours in 2019.

Community Uptake To Date (on Abel)





NLPL Core Architecture

- Jointly maintained project directory:
- discipline-specific software and data;
- largely parallel across Abel and Taito;
- access for NLPL users and associates;
- from 'power users' to MSc students;
- maybe 3–4 million core hours in 2019.

Community-Maintained Infrastructure

- 29 tools in 81 module files to date;
- 2 to 8 tbytes of data (Abel, Taito);
- large corpora, word embeddings, ...

For Example: Hyper-Parameter Search

```
#SBATCH -time=12:00:00
#SBATCH -nodes=1
#SBATCH -ntasks-per-node=8
#SBATCH -mem-per-cpu=4096M

module use -a /projects/nlpl/software/modulefiles;
module load nlpl-nltk/3.4/3.7 nlpl-gensim/3.7.2/3.7;
module load nlpl-scipy/201901/3.7 nlpl-pytorch/1.1.0/3.7;

{
  for wd in 50 100 200 300 400; do
    for cd in 5 10 20 40 80; do
      for do in 0 0.1 0.2 0.4; do
        echo python3 tagger.py -wd $wd -cd $cd -do $do;
      done;
    done;
  done
} | xargs -d \n -n 1 -P 8 -t sh -c;
```

For Example: Hyper-Parameter Search

```
#SBATCH -time=12:00:00
#SBATCH -nodes=1
#SBATCH -ntasks-per-node=8
#SBATCH -mem-per-cpu=4096M

module use -a /projects/nlpl/software/modulefiles;
module load nlpl-nltk/3.4/3.7 nlpl-gensim/3.7.2/3.7;
module load nlpl-scipy/201901/3.7 nlpl-pytorch/1.1.0/3.7;

{
  for wd in 50 100 200 300 400; do
    for cd in 5 10 20 40 80; do
      for do in 0 0.1 0.2 0.4; do
        echo python3 tagger.py -wd $wd -cd $cd -do $do;
      done;
    done;
  done
} | xargs -d \n -n 1 -P 8 -t sh -c;
```

For Example: Hyper-Parameter Search

```
#SBATCH -time=12:00:00
#SBATCH -nodes=1
#SBATCH -ntasks-per-node=8
#SBATCH -mem-per-cpu=4096M

module use -a /projects/nlpl/software/modulefiles;
module load nlpl-nltk/3.4/3.7 nlpl-gensim/3.7.2/3.7;
module load nlpl-scipy/201901/3.7 nlpl-pytorch/1.1.0/3.7;

{
  for wd in 50 100 200 300 400; do
    for cd in 5 10 20 40 80; do
      for do in 0 0.1 0.2 0.4; do
        echo python3 tagger.py -wd $wd -cd $cd -do $do;
      done;
    done;
  done
} | xargs -d \n -n 1 -P 8 -t sh -c;
```

For Example: Hyper-Parameter Search

```
#SBATCH -time=12:00:00
#SBATCH -nodes=1
#SBATCH -ntasks-per-node=8
#SBATCH -mem-per-cpu=4096M

module use -a /projects/nlpl/software/modulefiles;
module load nlpl-nltk/3.4/3.7 nlpl-gensim/3.7.2/3.7;
module load nlpl-scipy/201901/3.7 nlpl-pytorch/1.1.0/3.7;

{
  for wd in 50 100 200 300 400; do
    for cd in 5 10 20 40 80; do
      for do in 0 0.1 0.2 0.4; do
        echo python3 tagger.py -wd $wd -cd $cd -do $do;
      done;
    done;
  done
} | xargs -d \n -n 1 -P 8 -t sh -c;
```

For Example: Hyper-Parameter Search

```
#SBATCH -time=12:00:00
#SBATCH -nodes=1
#SBATCH -ntasks-per-node=8
#SBATCH -mem-per-cpu=4096M

module use -a /projects/nlpl/software/modulefiles;
module load nlpl-nltk/3.4/3.7 nlpl-gensim/3.7.2/3.7;
module load nlpl-scipy/201901/3.7 nlpl-pytorch/1.1.0/3.7;

{
  for wd in 50 100 200 300 400; do
    for cd in 5 10 20 40 80; do
      for do in 0 0.1 0.2 0.4; do
        echo python3 tagger.py -wd $wd -cd $cd -do $do;
      done;
    done;
  done
} | xargs -d \n -n 1 -P 8 -t sh -c;
```

For Example: Hyper-Parameter Search

```
#SBATCH -time=12:00:00
#SBATCH -nodes=1
#SBATCH -ntasks-per-node=8
#SBATCH -mem-per-cpu=4096M

module use -a /projects/nlpl/software/modulefiles;
module load nlpl-nltk/3.4/3.7 nlpl-gensim/3.7.2/3.7;
module load nlpl-scipy/201901/3.7 nlpl-pytorch/1.1.0/3.7;

{
  for wd in 50 100 200 300 400; do
    for cd in 5 10 20 40 80; do
      for do in 0 0.1 0.2 0.4; do
        echo python3 tagger.py -wd $wd -cd $cd -do $do;
      done;
    done;
  done
} | xargs -d \n -n 1 -P 8 -t sh -c;
```

Partially Automated Software Module Creation

/projects/nlpl/operations/python/pytorch.txt

```
#
#$ module load gcc/4.9.2 cuda/9.0
#$ module load nlpl-numpy/1.16.3/$dialect
#$ module load nlpl-scipy/201901/$dialect
#
torch
torchvision
torchtext
```

- Push modularization: small building blocks; many different versions;
- driven by user needs; never change installed module (reproducability).

Partially Automated Software Module Creation

/projects/nlpl/operations/python/pytorch.txt

```
#
#$ module load gcc/4.9.2 cuda/9.0
#$ module load nlpl-numpy/1.16.3/$dialect
#$ module load nlpl-scipy/201901/$dialect
#
torch
torchvision
torchtext
```

- Push modularization: small building blocks; many different versions;
- driven by user needs; never change installed module (reproducability).

Partially Automated Software Module Creation

/projects/nlpl/operations/python/pytorch.txt

```
#
#$ module load gcc/4.9.2 cuda/9.0
#$ module load nlpl-numpy/1.16.3/$dialect
#$ module load nlpl-scipy/201901/$dialect
#
torch
torchvision
torchtext
```

- Push modularization: small building blocks; many different versions;
- driven by user needs; never change installed module (reproducability).

```
for i in python2/2.7.10 python3/3.5.5 python3/3.7.0; do
  module purge; module load $i;
  /projects/nlpl/operation/python/initialize \
    --version 1.1.0 pytorch
done
```

Partially Automated Software Module Creation

/projects/nlpl/operations/python/pytorch.txt

```
#
#$ module load gcc/4.9.2 cuda/9.0
#$ module load nlpl-numpy/1.16.3/$dialect
#$ module load nlpl-scipy/201901/$dialect
#
torch
torchvision
torchtext
```

- Push modularization: small building blocks; many different versions;
- driven by user needs; never change installed module (reproducability).

```
for i in python2/2.7.10 python3/3.5.5 python3/3.7.0; do
  module purge; module load $i;
  /projects/nlpl/operation/python/initialize \
    --version 1.1.0 pytorch
done
```

The Resulting Module Definition

/projects/nlpl/software/modulefiles/nlpl-pytorch/1.1.0/3.7

```
#%Module1.0

set root "/projects/nlpl/software/"
set name "pytorch"
set version "1.1.0"
set base [string cat $root $name "/" $version]

module load intel/2019.0
module load openssl.intel/1_1_1
module load python3/3.7.0
module load gcc/4.9.2
module load cuda/9.0
module load nlpl-numpy/1.16.0/3.7
module load nlpl-scipy/201901/3.7

prepend-path PYTHONPATH $base/lib/python3.7/site-packages
prepend-path LD_LIBRARY_PATH $base/lib
prepend-path PATH $base/bin/3.7
```

The Resulting Module Definition

/projects/nlpl/software/modulefiles/nlpl-pytorch/1.1.0/3.7

```
#%Module1.0

set root "/projects/nlpl/software/"
set name "pytorch"
set version "1.1.0"
set base [string cat $root $name "/" $version]

module load intel/2019.0
module load openssl.intel/1_1_1
module load python3/3.7.0
module load gcc/4.9.2
module load cuda/9.0
module load nlpl-numpy/1.16.0/3.7
module load nlpl-scipy/201901/3.7

prepend-path PYTHONPATH $base/lib/python3.7/site-packages
prepend-path LD_LIBRARY_PATH $base/lib
prepend-path PATH $base/bin/3.7
```

The Resulting Module Definition

/projects/nlpl/software/modulefiles/nlpl-pytorch/1.1.0/3.7

```
#%Module1.0

set root "/projects/nlpl/software/"
set name "pytorch"
set version "1.1.0"
set base [string cat $root $name "/" $version]

module load intel/2019.0
module load openssl.intel/1_1_1
module load python3/3.7.0
module load gcc/4.9.2
module load cuda/9.0
module load nlpl-numpy/1.16.0/3.7
module load nlpl-scipy/201901/3.7

prepend-path PYTHONPATH $base/lib/python3.7/site-packages
prepend-path LD_LIBRARY_PATH $base/lib
prepend-path PATH $base/bin/3.7
```

The Resulting Module Definition

/projects/nlpl/software/modulefiles/nlpl-pytorch/1.1.0/3.7

```
#%Module1.0

set root "/projects/nlpl/software/"
set name "pytorch"
set version "1.1.0"
set base [string cat $root $name "/" $version]

module load intel/2019.0
module load openssl.intel/1_1_1
module load python3/3.7.0
module load gcc/4.9.2
module load cuda/9.0
module load nlpl-numpy/1.16.0/3.7
module load nlpl-scipy/201901/3.7

prepend-path PYTHONPATH $base/lib/python3.7/site-packages
prepend-path LD_LIBRARY_PATH $base/lib
prepend-path PATH $base/bin/3.7
```

Interim Summary: Typical NLP Computing

Throughput Limited Parallelization; Many Jobs to ‘Turn Around’;

Diversity Some Large-Memory Needs; Some I/O-Bound Jobs;

Interim Summary: Typical NLP Computing

Throughput Limited Parallelization; Many Jobs to ‘Turn Around’;

Diversity Some Large-Memory Needs; Some I/O-Bound Jobs;

Hardware Acute GPU Shortage in Norway (and, for now, Finland);

Interim Summary: Typical NLP Computing

Throughput Limited Parallelization; Many Jobs to ‘Turn Around’;

Diversity Some Large-Memory Needs; Some I/O-Bound Jobs;

Hardware Acute GPU Shortage in Norway (and, for now, Finland);

Few Privacy Concerns Predominantly Public, Non-Sensitive Data;

Interim Summary: Typical NLP Computing

Throughput Limited Parallelization; Many Jobs to 'Turn Around';

Diversity Some Large-Memory Needs; Some I/O-Bound Jobs;

Hardware Acute GPU Shortage in Norway (and, for now, Finland);

Few Privacy Concerns Predominantly Public, Non-Sensitive Data;

User Communities Discipline-Specific Installations & Support;

Self-Help Large, common data sets; specialized, in-house software;

Internationalization (At Least Nordic) Cross-Border Perspective.

Interim Summary: Typical NLP Computing

Throughput Limited Parallelization; Many Jobs to 'Turn Around';

Diversity Some Large-Memory Needs; Some I/O-Bound Jobs;

Hardware Acute GPU Shortage in Norway (and, for now, Finland);

Few Privacy Concerns Predominantly Public, Non-Sensitive Data;

User Communities Discipline-Specific Installations & Support;

Self-Help Large, common data sets; specialized, in-house software;

Internationalization (At Least Nordic) Cross-Border Perspective.

Challenges After Two Years of NLPL

- For now, virtual laboratories on Abel and Taito expensive to migrate;
- want automated instantiation (across platforms) of NLPL 'blueprint'.

Community Formation: Annual Winter School

Community Formation: Annual Winter School

Credits: We are Especially Grateful To

Credits: We are Especially Grateful To

NeIC: The Nordic e-Infrastructure Collaboration;

Credits: We are Especially Grateful To

NeIC: The Nordic e-Infrastructure Collaboration;

CSC (Finland) and Uninett Sigma2 (Norway);

Credits: We are Especially Grateful To

NeIC: The Nordic e-Infrastructure Collaboration;

CSC (Finland) and Uninett Sigma2 (Norway);

The Project Partners and NLPL Team;

Credits: We are Especially Grateful To

NeIC: The Nordic e-Infrastructure Collaboration;

CSC (Finland) and Uninett Sigma2 (Norway);

The Project Partners and NLPL Team;

The Nordic and European Tax Payers;

Credits: We are Especially Grateful To

NeIC: The Nordic e-Infrastructure Collaboration;

CSC (Finland) and Uninett Sigma2 (Norway);

The Project Partners and NLPL Team;

The Nordic and European Tax Payers;

Bjørn Lindi, our Project Manager.