Code is Big Data

Emmanuel Bengio - COMP 762

- D3, Data-Driven Documents
- Learning from Examples to Improve Code Completion Systems
- Method-Call Recommendations from Implicit Developer Feedback
- Milepost GCC: Machine Learning Enabled Self-tuning Compiler
- Learning to Execute

D^3, Data-Driven Documents

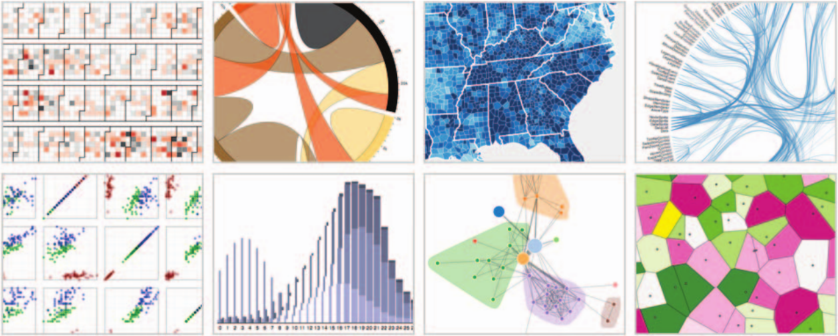
Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer

IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, 2011

- Data visualization library
- Instead of creating a whole new hidden representation of the data, it operates directly on the DOM
- Bind data to some html/svg element, apply explicit transformations on it (e.g. position, color, thickness)
- Visualization is important in many ways to help us make sense of data (and we should do alot more of it)

D^3, Data-Driven Documents

- Also looks really nice



Learning from Examples to Improve Code Completion Systems

Marcel Bruch, Martin Monperrus, and Mira Mezini
ESEC 2009

- Comparison of 3 non-trivial code completion systems:
  - frequency based
  - association rule based
  - *Best Matching Neighbor* (kNN)
- Quantitative evaluation
- Evaluation by 10 experienced users
  $\rightarrow$ kNN-like approach performs best

Learning from Examples to Improve Code Completion Systems

- There are gigabytes of code data (github, bitbucket, etc.)
- There are usage patterns, a basic algorithm like kNN does something good!
- (Imagine if we threw Deep Learning at this problem)

Method-Call Recommendations from Implicit Developer Feedback

Sven Amann, Sebastian Proksch, and Mira Mezini
ICSE 2014

- Use history of user's code completions as data
- Pose problem as Collaborative Filtering
- For new code completions, find best matches using CF
- The data is there, new CF algorithms are invented every year

Milepost GCC: Machine Learning Enabled Self-tuning Compiler

Grigori Fursin, Yuriy Kashnikov, Abdul Wahid Memon, Zbigniew Chamski, Olivier Temam, Mircea Namolaru, Elad Yom-Tov, Bilha Mendelson, Ayal Zaks, Eric Courtois, Francois Bodin, Phil Barnard, Elton Ashton, Edwin Bonilla, John Thomson, Christopher K. I. Williams, Michael O'Boyle

International journal of parallel programming, 2011

- Learn mapping from AST features to "-Ox -f[no]y -f..."
- Use machine learning to learn the mapping:
  - Predictive Search Distributions
  - Decision Trees
- Reduce execution time (~17%)
- Reduce compilation time (12%) and code size (7%)

Learning To Execute

Wojciech Zaremba, and Ilya Sutskever

- Learn mapping from source code to output (at character level)

Input:
```
j=8584
for x in range(8):
  j+=920
b=(1500+j)
print((b+7567))
```
Target: 25011.

Input:
```
i=8827
c=(i-5347)
print((c+8704) if 2641<8500 else 5308)
```
Target: 12184.

- Learned model performs 9-digit addition with 99% accuracy

Learning To Execute

- Why is this interesting?
  - Relatively simple RNN model can do this
  - Code is a viable domain for NN
- Using this kind of model for my last 3 papers could be beneficial:
  - Representation learning
  - Generative prediction (code completion)
  - Predictive supervised model