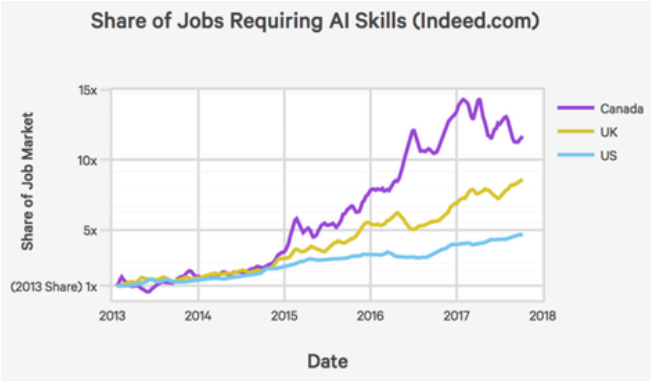
Non-Recursive Code Making Software

Kimberly Lo (kimberll@andrew.cmu.edu), Lisa Lo (llo1@andrew.cmu.edu)

Carnegie Mellon University

Background

- Code that creates other functioning programs that solves a different problem than what was directly programmed
- Examples: AI coding games, AutoML



Positives of AI

- Can create programs better than we can create it
- Can be used to improve economic equity through accessibility

Negatives of AI

- Lack of understanding
  -> inability to be transparent
  -> potential for bugs we can't fix

Carnegie Mellon University
School of Computer Science

Superintelligence - Sci-Fi or Reality?

Positive

- Solve problems we can't tackle
- Don't need coders – just testers?
- Project 2045

Negative

- Dangerous
- Making bigger problems than we can solve

Our Position

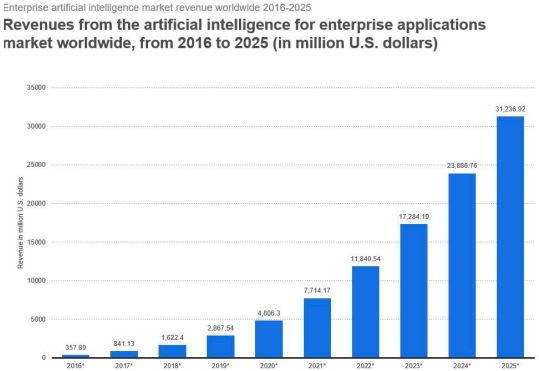
- We support development of code that creates code non-recursively

Conditions:

- Accessibility for all education levels
- Rigorous testing before release
- Backup system for emergencies

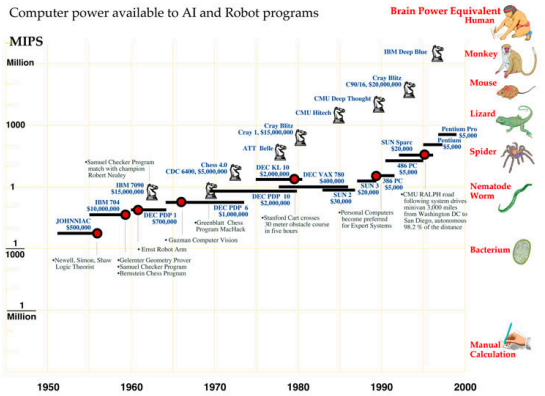
Consequentialist & Virtue Ethics Perspectives:

- Making AI a tool rather than a skill so that anyone can use this technology
- Minimize dangers of how we approach superintelligence



Implications

- New avenues of creativity
- Higher productivity
- Increased equality
- More advanced levels of machine learning intelligence
- Potentially hard to maintain
- Need to redefine responsibility among involved parties



Conclusion

- Key is to minimize problems that are bound to occur
  - Plan for necessary policies to keep users safe
- Creating AI that creates code non-recursively is a step towards establishing more intelligent programs which is desired

Carnegie Mellon University