Does AI Replace Software Developers?

Michael Pradel (University of Stuttgart)

Based on joint work with Aryaz Eghbali, Islem Bouzenia, Luca Di Grazia, and Wai Chow
Will AI Replace Software Engineers?

Mohammed Hassan • Follow
3 min read • Feb 25
Will AI Replace Software Engineers?

The Future Of Programming: Will AI Replace Programmers?

Mohammed Has
3 min read · Feb 2

Strivemindz
Your Digital Transformation Partner
Published Jun 5, 2023
Will AI Replace Software Engineers?

ChatGPT may be coming for our jobs. Here are the 10 roles that AI is most likely to replace.

Aaron Mok and Jacob Zinicola  Updated Sep 4, 2023, 4:24 PM GMT+2

Insider compiled a list of the 10 jobs that could be disrupted by AI tools like ChatGPT, according to experts. Jens Schluter/Getty Images
ChatGPT may be coming for our jobs. Here are the 10 roles that AI is most likely to replace.

Aaron McK and Jacob Zinkula  Updated Sep 4, 2023, 4:24 PM GMT+2

LIBRARY / AI IN SOFTWARE DEVELOPMENT /

Is There a Future for Software Engineers? The Impact of AI [2023]

Let's investigate how AI can shape software development, which skills will be relevant in the nearest future, and how to approach all those changes.

10 min read – Last updated on September 18, 2023
This Talk

1) Overview of state-of-the-art
2) My answer to the question
3) Peek into the future
Work on AI-based software dev. tools *

* Estimate based on *Neural Software Analysis*, Pradel & Chandra, CACM’22
Timeline

Work on AI-based software dev. tools *

Today: $\approx 50\%$ of papers at top SE conferences

2015

TabNine

2020

Copilot

ChatGPT

* Estimate based on *Neural Software Analysis*, Pradel & Chandra, CACM’22
Timeline

Work on AI-based software dev. tools *

Today: ≈ 50% of papers at top SE conferences

Our first publication (“Deep Learning to Find Bugs”)

TabNine, Copilot, ChatGPT

2015

* Estimate based on Neural Software Analysis, Pradel & Chandra, CACM’22
Three Examples

Software development activities (traditionally done by hand):

- Write code
- Search for bugs
- Fix bugs
Three Examples

Software development activities (traditionally done by hand):

- Write code
- Search for bugs
- Fix bugs
- Code completion
- Neural bug detection
- Automated program repair
Code Completion

- [Copilot demo]
Problem Solved?

Useful, but many unsolved questions

- Challenge 1: Project-specific APIs
- Challenge 2: Prioritizing context
Problem Solved?

Useful, but many unsolved questions

- Challenge 1: Project-specific APIs
- Challenge 2: Prioritizing context

Code used for training
Problem Solved?

Useful, but many unsolved questions

- Challenge 1: Project-specific APIs
- Challenge 2: Prioritizing context

Code used for training  AI  Your project
Problem Solved?

Useful, but many unsolved questions

- **Challenge 1**: Project-specific APIs
- **Challenge 2**: Prioritizing context

Prompt size of models: < Size of real-world projects:
An API somewhere in our project:

```python
def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)
```

Code we want to complete:

```python
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
```
An API somewhere in our project:

```python
def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)
```

Code we want to complete:

```python
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
    return sorted(docs, key=lambda doc: doc.relevance, reverse=True)[:top_k]
```

Prediction by CodeGen model
An API somewhere in our project:

```python
def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)
```

Code we want to complete:

```python
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
    docs_scores = [(doc, compute_relevance_score(doc, keyword) ) for doc in docs]
    sorted_docs_scores = sorted(docs_scores, key=lambda x: x[1], reverse=True)
    return [doc_score[0] for doc_score in sorted_docs_scores[:top_k]]
```

Prediction by ChatGPT’s model
An API somewhere in our project:

```python
def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)
```

Code we want to complete:

```python
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
    docs_scores = [(doc, compute_relevance_score(doc, keyword)) for doc in docs]
    sorted_docs_scores = sorted(docs_scores, key=lambda x: x[1], reverse=True)
    return [doc_score[0] for doc_score in sorted_docs_scores[:top_k]]
```

Prediction by ChatGPT’s model

Problem: Hallucination
De-Hallucinator

Diagram:
- Code before completion location
- Initial prompt (type 1) to Large language model
- Augmented prompt (type 2)
- Augmented prompt (type 3)
- Retrieval of related APIs
- Prompt construction
- API references
- Static pre-analysis
- Index of APIs
- Completion
An API somewhere in our project:

def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)

Code we want to complete:

def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
An API somewhere in our project:

```python
def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)
```

Code we want to complete:

```python
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
    return sorted(docs, key=lambda doc: doc.relevance, reverse=True)[:-top_k]
```
An API somewhere in our project:

```python
def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)
```

Code we want to complete:

```python
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
    return sorted(docs, key=lambda doc: doc.relevance, reverse=True)[:top_k]
```

Augmented prompt:

```python
# API Reference:
# relevance(document: str, keyword: str) -> float # Returns the relevance of the document to the keyword.
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
```
An API somewhere in our project:

```python
def relevance(document: str, keyword: str) -> float:
    """Returns the relevance of the document to the keyword."""
    return document.count(keyword) / len(document)
```

Code we want to complete:

```python
def search(ds: DataStore, keyword: str, top_k: int) -> List[str):
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
    return sorted(docs, key=lambda doc: doc.relevance, reverse=True)[:top_k]
```

Augmented prompt:

```python
# API Reference:
# relevance(document: str, keyword: str) -> float # Returns the relevance of the document to the keyword.
def search(ds: DataStore, keyword: str, top_k: int) -> List[str]:
    """Returns the top_k most relevant documents that contain the keyword sorted by relevance."""
    docs = ds.find_by_keyword(keyword)
    return sorted(docs, key=lambda doc: doc.relevance(keyword, doc), reverse=True)[:top_k]
```
Three Examples

Software development activities (traditionally done by hand):

- Write code
- Search for bugs
- Fix bugs
Three Examples

Software development activities (traditionally done by hand):

- Write code
- Search for bugs
- Fix bugs

Does AI replace software developers?

Powerful tool, but (so far) only for small-scale code completion
**Motivation**

**Example 1:**

```python
if len(bits) != 4 or len(bits) != 6:
    raise template.TemplateSyntaxError("%r takes exactly four or six arguments (second argument must be 'as')" % str(bits[0]))
```
Motivation

Example 1: Always True

```python
if len(bits) != 4 or len(bits) != 6:
raise template.TemplateSyntaxError("%r takes exactly four or six arguments (second argument must be ‘as’)" % str(bits[0]))
```

Doesn't match the message
Motivation

Example 1:  

```python
if len(bits) != 4 or len(bits) != 6:
    raise template.TemplateSyntaxError("%r takes exactly four or six arguments (second argument must be 'as')" % str(bits[0]))
```

Example 2:

```python
if n2 > n1 :
    raise ValueError('Total internal reflection impossible for n1 > n2')
```
Motivation

Example 1: Always True

```python
if len(bits) != 4 or len(bits) != 6:
    raise template.TemplateSyntaxError("%r takes exactly four or six arguments (second argument must be 'as')" % str(bits[0]))
```

Example 2: Doesn’t match the message

```python
if n2 > n1:
    raise ValueError('Total internal reflection impossible for n1 > n2')
```

Condition and message are inconsistent

Always True Doesn’t match the message

Condition and message are inconsistent
Goal:

Detect _condition-message_ inconsistencies

- Why?
  - Incorrect conditions may raise unnecessary warnings or suppress expected warnings
  - Incorrect messages make debugging unnecessarily hard

- Hard problem!
  - Must understand both NL and PL
Overview of CMI-Finder

- Code corpus
- Code to analyze
- Data extraction
- Message-condition pairs
- Preprocessing & embedding
- Neural model
- Warnings about inconsistencies

Generate inconsistent examples

6x

Training  

Prediction
Does It Work?

- 78% precision and 72% recall on historic bugs
- 50 new inconsistencies in previously unseen projects
- Complements traditional linters
Does It Work?

- **78% precision and 72% recall** on historic bugs
- **50 new inconsistencies** in previously unseen projects
- **Complements traditional linters**

**But:** Limited to a specific kind of bug
Software development activities (traditionally done by hand):

- Write code
- Search for bugs
- Fix bugs
Three Examples

Software development activities (traditionally done by hand):

■ Write code
■ Search for bugs
■ Fix bugs

Does AI replace software developers?

Makes bug detection easier, but human judgement still required
Types in Python

Typical evolution of a Python project:

Code without type annotations

```python
def f(x, y):
    s = x + y
    if (s % 2) == 0:
        return True
```

The Evolution of Type Annotations in Python: An Empirical Study, FSE’22
Types in Python

Typical evolution of a Python project:

Partially annotated code

```python
def f(x: int, y) -> bool:
    s: int = x + y
    if (s % 2) == 0:
        return True
```

The Evolution of Type Annotations in Python: An Empirical Study, FSE’22
Types in Python

Typical evolution of a Python project:

Partially annotated code

```python
def f(x: int, y) -> bool:
s: int = x + y
if (s % 2) == 0:
    return True
```

Type error!
Typical evolution of a Python project:

```python
def f(x: int, y) -> Optional[bool]:
s: int = x + y
if (s % 2) == 0:
    return True
```

Fixed type error
Too Many Type Errors

- Most existing Python code bases:
  Plenty of static type errors
- Easy to detect by gradual type checker
- But: No time to fix them all
PyTy: Approach

Commits with type error fixes → Type checking & delta debugging → Dataset of isolated fixes

Code with type errors → PyTy model → Candidate fix → Fixed code

Fixed code → Type checking for validation

PyTy model → Fine-tuning → Pre-trained TFix model
## Does It Work?

<table>
<thead>
<tr>
<th>Classes of type errors</th>
<th>Samples (test set)</th>
<th>Effectiveness of PyTy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Error removal</td>
</tr>
<tr>
<td>Incompatible variable type</td>
<td>821 (83)</td>
<td>90.4%</td>
</tr>
<tr>
<td>Incompatible parameter type</td>
<td>600 (60)</td>
<td>80.0%</td>
</tr>
<tr>
<td>Incompatible return type</td>
<td>296 (30)</td>
<td>73.3%</td>
</tr>
<tr>
<td>Invalid type</td>
<td>291 (30)</td>
<td>100.0%</td>
</tr>
<tr>
<td>Unbound name</td>
<td>258 (26)</td>
<td>76.9%</td>
</tr>
<tr>
<td>Incompatible attribute type</td>
<td>258 (26)</td>
<td>92.3%</td>
</tr>
<tr>
<td>Unsupported operand</td>
<td>124 (13)</td>
<td>76.9%</td>
</tr>
<tr>
<td>Strengthened precondition</td>
<td>59 (6)</td>
<td>83.3%</td>
</tr>
<tr>
<td>Weakened postcondition</td>
<td>51 (6)</td>
<td>50.0%</td>
</tr>
<tr>
<td>Call error</td>
<td>8 (1)</td>
<td>100.0%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>2,766 (281)</strong></td>
<td><strong>85.4%</strong></td>
</tr>
</tbody>
</table>
Examples

Code with **type error**:

```python
vprint(f"{prefix} {lineno}: {action_name}
Constrain Mouse: {'yes' if constraint > 0
else ('no' if constrained == 0 else 'check stack')}}")
```

*Unbound name*

PyTy finds **exactly the developer fix**:  

```python
vprint(f"{prefix} {lineno}: {action_name}
Constrain Mouse: {'yes' if constraint > 0
else ('no' if constraint == 0 else 'check stack')}}")
```
Examples

Code with **type error**: Declared to have type `str` but used as `bytes`

```python
string = _fmt(string)
return lib.TCOD_console_get_height_rect_fmt(
    self.console_c, x, y, width, height, string
)
```

**PyTy finds a valid fix:**

```python
byte_string = _fmt(string)
return lib.TCOD_console_get_height_rect_fmt(
    self.console_c, x, y, width, height, byte_string
)
```

**Developer fix (semantically equivalent):**

```python
return lib.TCOD_console_get_height_rect_fmt(
    self.console_c, x, y, width, height, _fmt(string)
)
```
Three Examples

Software development activities (traditionally done by hand):

- Write code
- Search for bugs
- Fix bugs
Three Examples

Software development activities (traditionally done by hand):
  - Write code
  - Search for bugs
  - Fix bugs

Does AI replace software developers?

Successfully automates bug fixing for a specific class of bugs
Current **AI-based development tools** are only the beginning!

- **Autonomous software development agents**
- **Neuro-symbolic program analysis**
Big Picture

Key feature of humans:
Ability to develop tools

Software development tools
Big Picture

Key feature of humans:
Ability to develop tools

Traditionally:
Compilers and hand-crafted program analyses

Now:
Learning-based tools
Does AI replace software developers?
Does AI replace software developers?

Yes, but only those who don’t adapt.