Bogdan Vasilescu
(CMU, ISR)
@b_vasilescu

Casey Casalnuovo
(UCDavis)

Prem Devanbu
(UCDavis)
@devanbu

Recovering Clear, Natural Identifiers from Obfuscated (JavaScript) Names

STRUDEL
SOCIO-TECHNICAL RESEARCH USING DATA EXCAVATION LAB

isr institute for SOFTWARE RESEARCH

Carnegie Mellon University

UCDAVIS
UNIVERSITY OF CALIFORNIA

NSF

Today

```
var geom2d = function() {
  var t = numeric.sum;
  function r(n, r) {
    this.x = n;
    this.y = r;
  }
  u(r, {
    P: function e(n) {
      return t([ this.x * n.x,
                 this.y * n.y ]);
    }
  });
  function u(n, r) {
    for (var t in r) n[t] = r[t];
    return n;
  }
  return {
    V: r
  };
}();
```

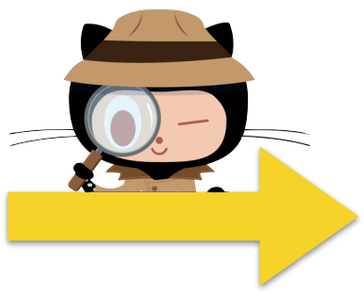
Today

```
var geom2d = function() {
  var t = numeric.sum;
  function r(n, r) {
    this.x = n;
    this.y = r;
  }
  u(r, {
    P: function e(n) {
      return t([ this.x * n.x,
                 this.y * n.y ]);
    }
  });
  function u(n, r) {
    for (var t in r) n[t] = r[t];
    return n;
  }
  return {
    V: r
  };
}();
```

Today

Data-driven method + tool

```
var geom2d = function() {
  var t = numeric.sum;
  function r(n, r) {
    this.x = n;
    this.y = r;
  }
  u(r, {
    P: function e(n) {
      return t([ this.x * n.x,
                 this.y * n.y ]);
    }
  });
  function u(n, r) {
    for (var t in r) n[t] = r[t];
    return n;
  }
  return {
    V: r
  };
}();
```



```
var geom2d = function() {
  var sum = numeric.sum;
  function Vector2d(x, y) {
    this.x = x;
    this.y = y;
  }
  mix(Vector2d, {
    P: function dotProduct(vector) {
      return sum([ this.x * vector.x,
                   this.y * vector.y ]);
    }
  });
  function mix(dest, src) {
    for (var k in src) dest[k] = src[k];
    return dest;
  }
  return {
    V: Vector2d
  };
}();
```

Why?

- Programs are (also) written to be read

"Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on ***explaining to human beings what we want a computer to do****."* [Don Knuth]

Why?

- Programs are (also) written to be read
- Well-chosen variable names are critical to source code **readability**, **reusability**, **maintainability**
- Example tasks:
  - reverse engineering binaries
  - reverse engineering obfuscated JavaScript
  - consistent styling in large, distributed teams

Why?

- Programs are (also) written to be read
- Well-chosen variable names are critical to source code **readability**, **reusability**, **maintainability**
- Example tasks:
  - reverse engineering binaries
  - reverse engineering obfuscated JavaScript
  - consistent styling in large, distributed teams

Why?

- Programs
- Well-chos
  code **rea**
- Example
  - reverse
  - reverse
  - consiste

JSNice.org

[Predicting program properties from Big Code, ACM POPL 2015]

✓ Every country ✓ 160,000 users ✓ Top ranked tool

This Page Amsterdam @thispage_ams · Jul 16
Do you write ugly JavaScript code? Not to worry. JSNice will make it look like you are a superstar coder. Yai! - buff.ly/1HR4JL7

Ingvar Stepanyan @RReverser · Aug 6
JSNice.org became my must-have tool for code deobfuscation.
Expand Reply Retweet Favorite More

Brevity @seekbrevity · Jul 28
JSNice is an a amazing tool for de-minifying #javascript files. JSNice.org, Its great for #learning and reverse engineering.
Expand Reply Retweet Favorite More

Alvaro Sanchez @alvasavi · Jun 10
This is gold.
Statistical renaming, Type inference and Deobfuscation.
jsnice.org
Expand Reply Retweet Favorite More

Alex Vanston @mrvdot · Jun 7
I've been looking for this for years: JS NICE buff.ly/1pQ5qfr #javascript #unminify #deobfuscate #makeItReadable
Expand Reply Retweet Favorite More

Kamil Tomšík @cztomsik · Jun 6
tell me how this works!
de-minify #jquery #javascript incl. args, vars & #jsdoc
impressive! jsnice.org

CodeGeekz

COMMENTS
Frici Komesz on: 20 Online Code Editors and Tools for Developers

HOME WORDPRESS WEB DESIGN TYPOGRAPHY RESOURCES ARCHIVES CONTACT ADVERTISE

20 Essential Tools for Coders

BY GAVIN IN DEVELOPMENT RESOURCES – 28 JUL, 2014

20 Best Freebies for Web Designers of Year 2014

JavaScript迷宮救星 — JS nICE

JS NICE

right place, I have gathered 20 Ess 1) jsNi
development tasks. Following dev
beautify, store your snippet and h
you will find the list handy and acc

1. JS Nice

JsNice is tool that help you to make even make even obfuscated JavaScript code readable. JSnice is Statical Re-naming, Type Inference and De obfuscation.

Martin Vechev, “Probabilistic Learning From Big Code”. Keynote at ISSTA 2016

Key ingredient

- The “naturalness” of software [Hindle et al, 2011]

Natural languages are complex

Natural languages are complex

Tiger, Tiger
burning bright
In the forests
of the night
What immortal
hand or eye,
Could frame
thy fearful
symmetry?

..but most utterances are simple & repetitive

English, தமிழ், German

Can be Rich, Powerful, Expressive

English, தமிழ், German

Can be Rich, Powerful, Expressive

..but “in nature” is mostly Simple, Repetitive, Boring

English, தமிழ், German

Can be Rich, Powerful, Expressive

..but “in nature” is mostly Simple, Repetitive, Boring

Statistical Models

English, தமிழ், German

Can be Rich, Powerful, Expressive

..but “in nature” is mostly Simple, Repetitive, Boring

Statistical Models

The “naturalness of software” thesis

Programming Languages are complex...

...but Natural Programs are simple & repetitive.

and this, too, CAN BE EXPLOITED!!

[Hindle et al, 2011]

Google Translate

Translate

Turn off instant translation

English | Persian | Chinese | English - detected

Romanian | English | Dutch

Translate

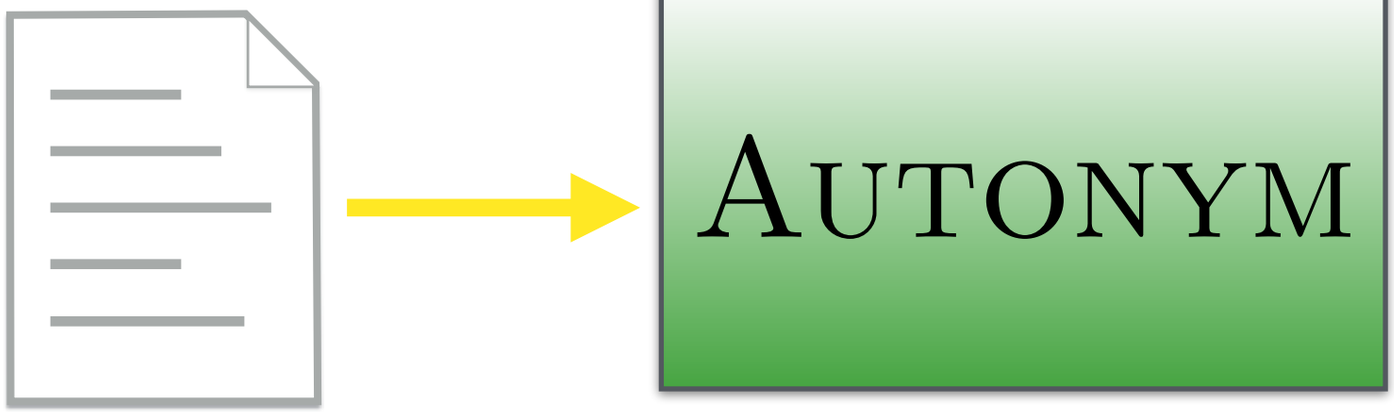
I know what you named your identifiers last summer!

Ik weet wat je je id's genaamd afgelopen zomer!

JSNaughty.org

Autonym

JSNaughty.org



Minified Source Code

```
function u(n, r) {
    for (var t in r) n[t] = r[t];
    return n;
}
```

JSNaughty.org

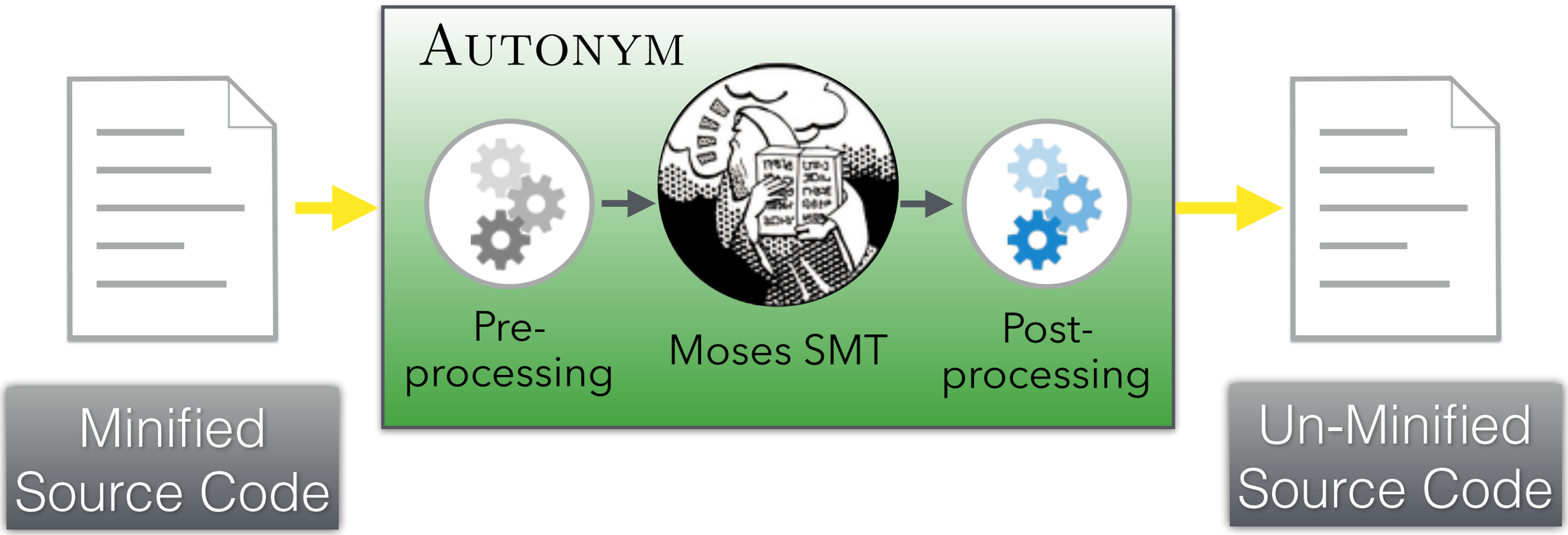
Minified Source Code

```
function u(n, r) {
    for (var t in r) n[t] = r[t];
    return n;
}
```

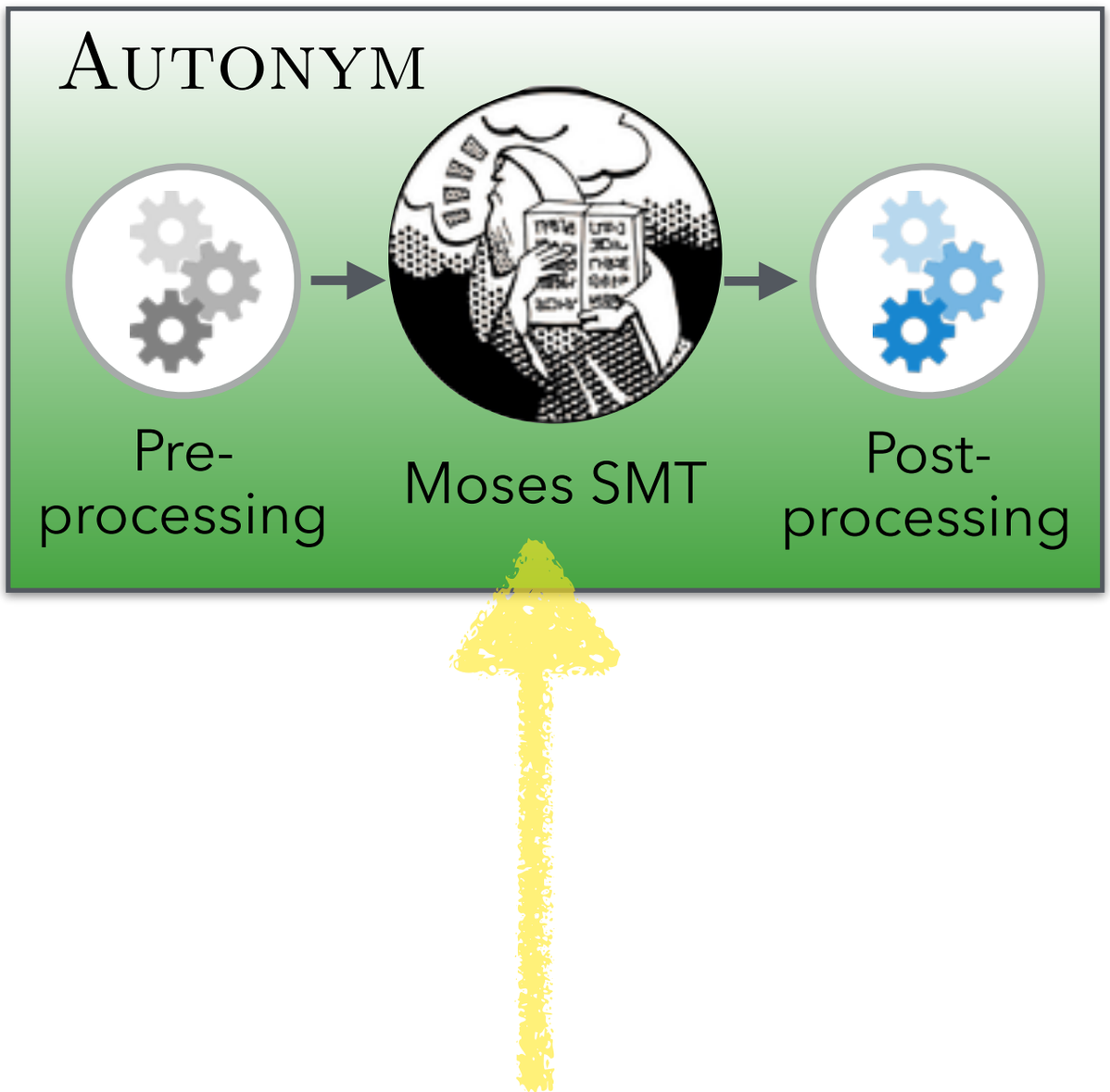
Un-Minified Source Code

```
function mix(dest, src) {
    for (var k in src) dest[k] = src[k];
    return dest;
}
```

JSNaughty.org



JSNaughty.org



What's the relevance of Machine Translation?

Noisy channel translation model

Noisy channel translation model

Noisy channel translation model

Noisy channel translation model

$p(f \mid e)$

channel model

Noisy channel translation model

$p(e)$
language model

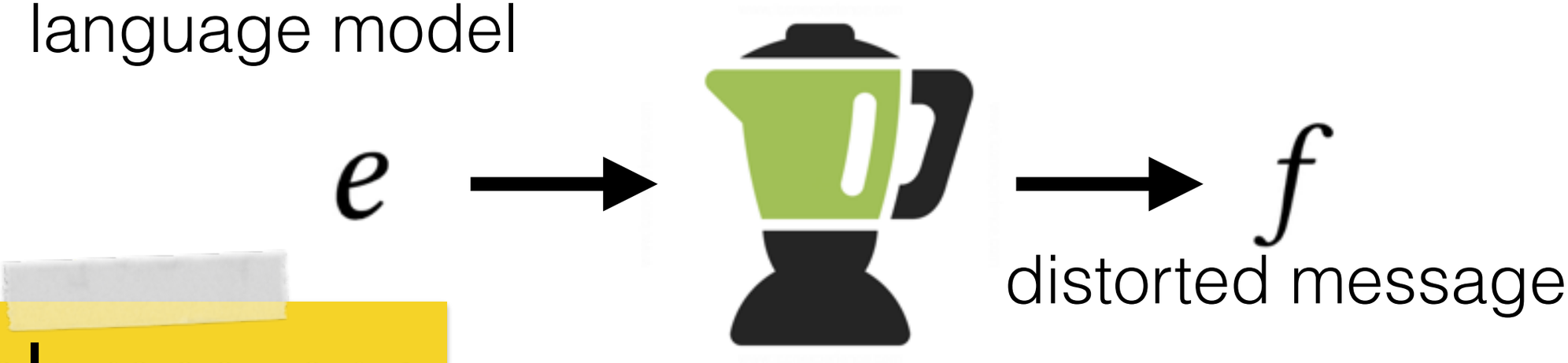
$p(f \mid e)$
channel model

distorted message

Noisy channel translation model

$p(e)$
language model

$p(f \mid e)$
channel model



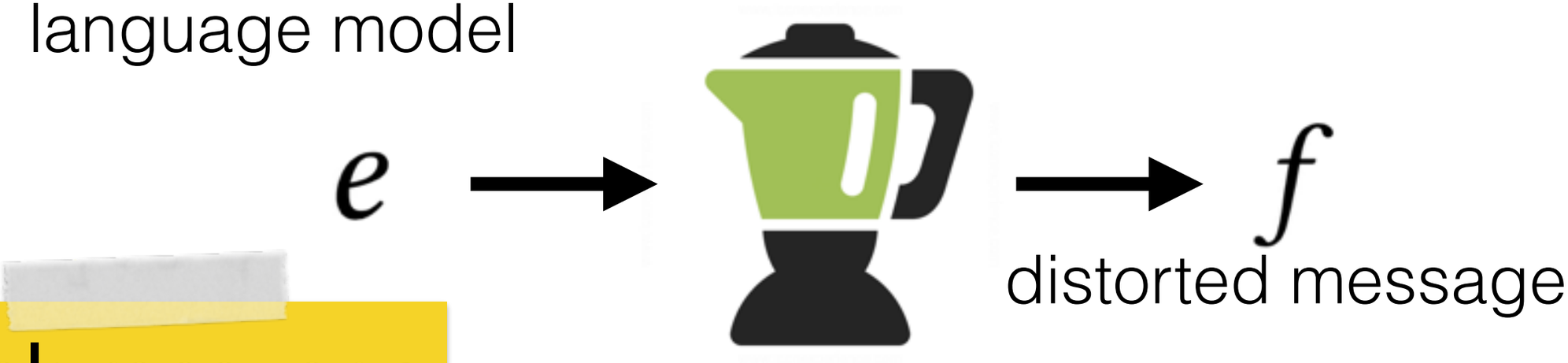
distorted message

Goal: recover e

Noisy channel translation model

$p(e)$
language model

$p(f \mid e)$
channel model



distorted message

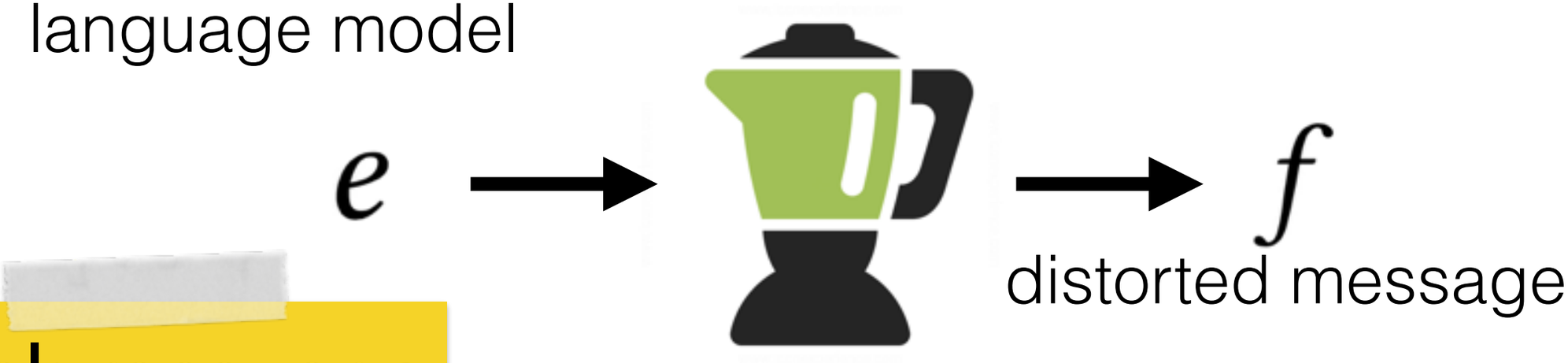
Goal: recover e

$$e_{\text{best}} = \underset{e}{\text{argmax}}\, p(e \mid f)$$

Noisy channel translation model

$p(e)$
language model

$p(f \mid e)$
channel model



Goal: recover e

$$e_{\text{best}} = \underset{e}{\operatorname{argmax}}\, p(e \mid f)$$

Bayes theorem

$$= \underset{e}{\operatorname{argmax}}\, \frac{p(f \mid e)p(e)}{p(f)}$$

$$= \underset{e}{\operatorname{argmax}}\, p(f \mid e)p(e) \quad \text{(for a given } f\text{)}$$

Noisy channel translation model

$p(e)$
language model

$p(f \mid e)$
channel model

Goal: recover e

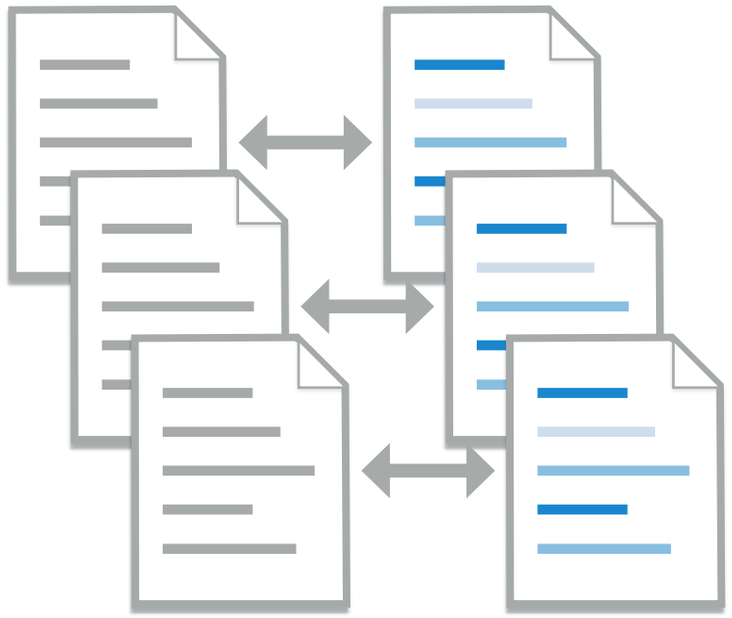
$$e_{\text{best}} = \underset{e}{\operatorname{argmax}}\, p(f \mid e)p(e)$$

Translation (channel distortion) model

Language model

Translating French (f) to English (e)

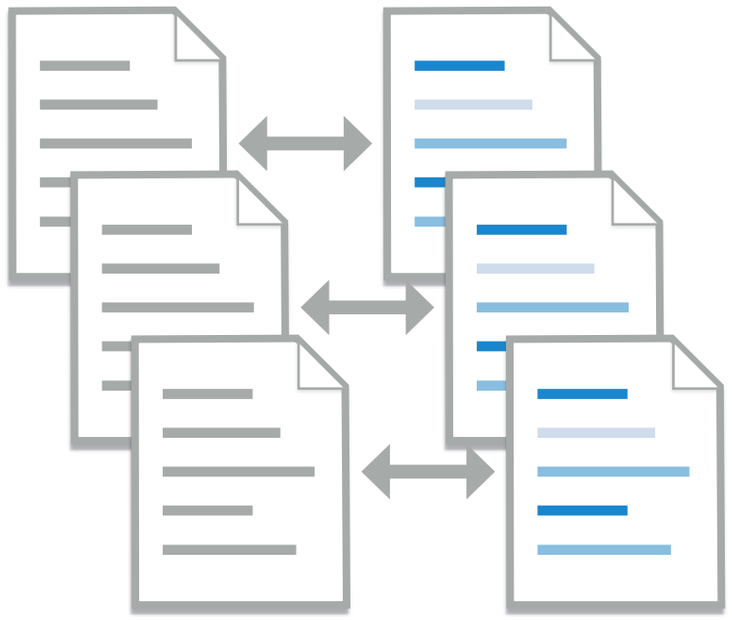
Translating French (f) to English (e)



Aligned French-English Corpus

$$p(f \mid e)$$

Translating French (f) to English (e)



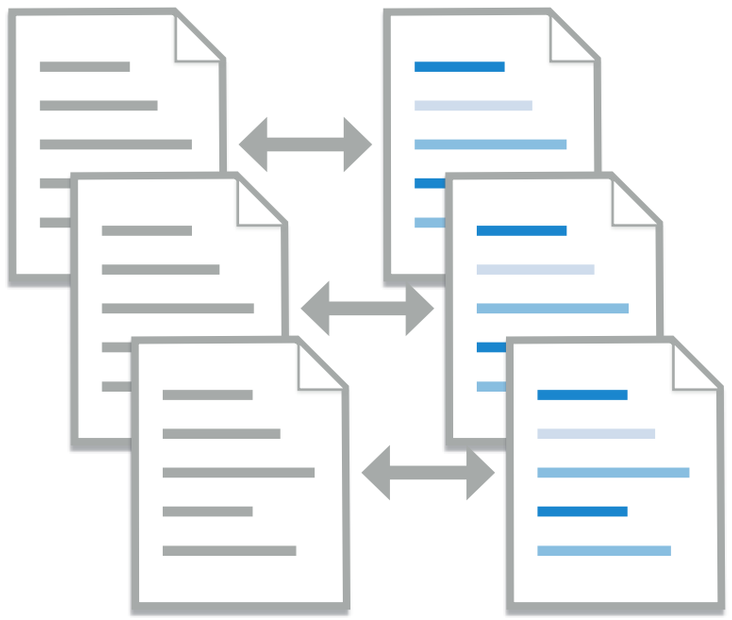
Aligned French-English Corpus

$p(f \mid e)$

English Corpus

$p(e)$

Translating French (f) to English (e)



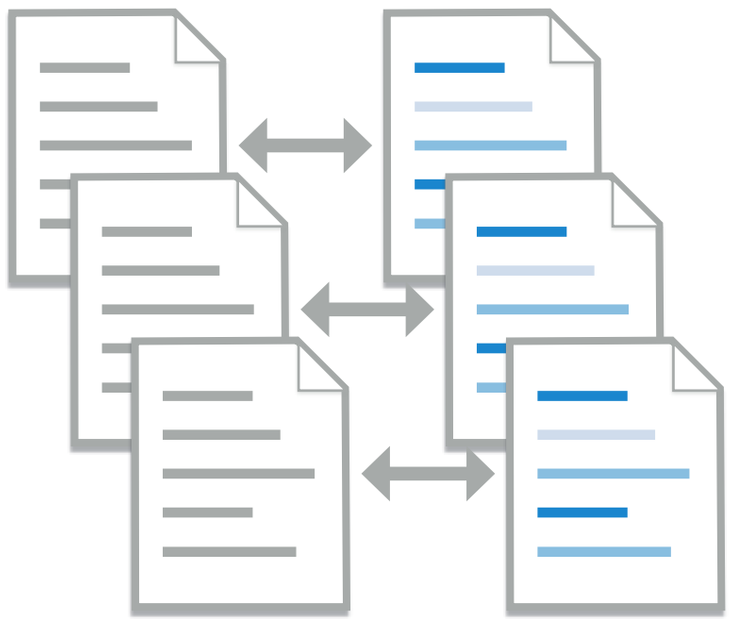
Aligned French-English Corpus

$$p(f \mid e)$$

English Corpus

$$p(e)$$

Translating minified (f) to clear JS (e)



Aligned Clear-Minified Code Corpus

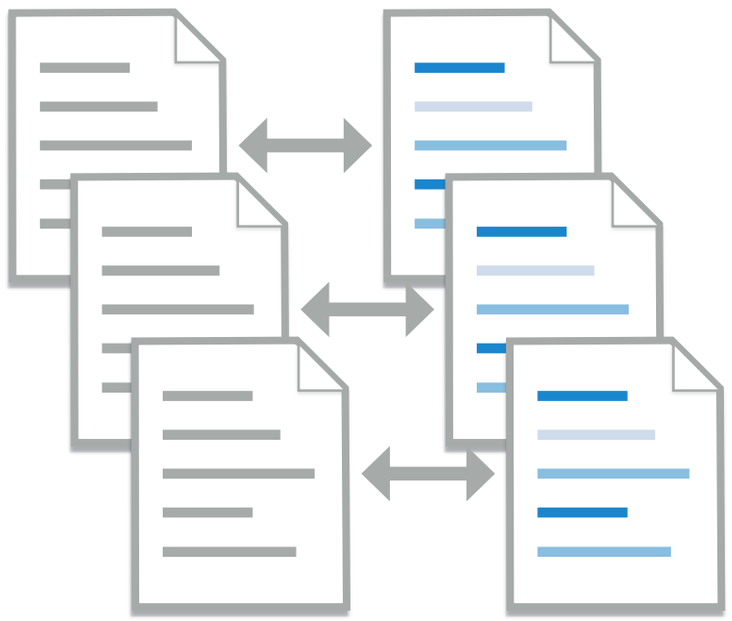
Clear Code Corpus

$p(f \mid e)$

$p(e)$

Translating minified (f) to clear JS (e)

GitHub + minifier



Aligned Clear-Minified Code Corpus

Clear Code Corpus

$p(f \mid e)$

$p(e)$

MOSES
statistical machine translation system

Alignment

EN: I know what you named your identifiers!

NL: Ik weet wat je je ID's genoemd!

Natural language:
non-trivial alignment

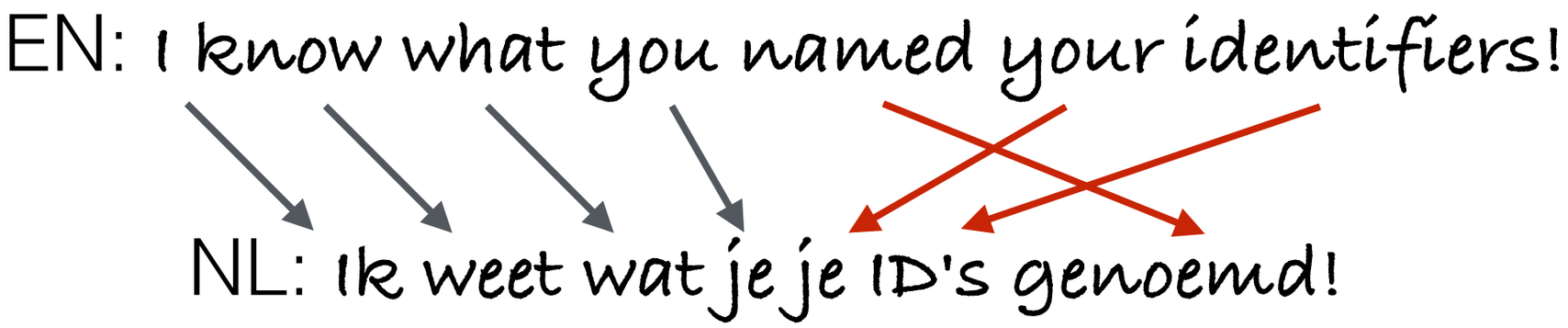
- Reordering
- Different length
- Dropped words

Alignment

Natural language: non-trivial alignment

- Reordering
- Different length
- Dropped words

Alignment



Natural language: non-trivial alignment

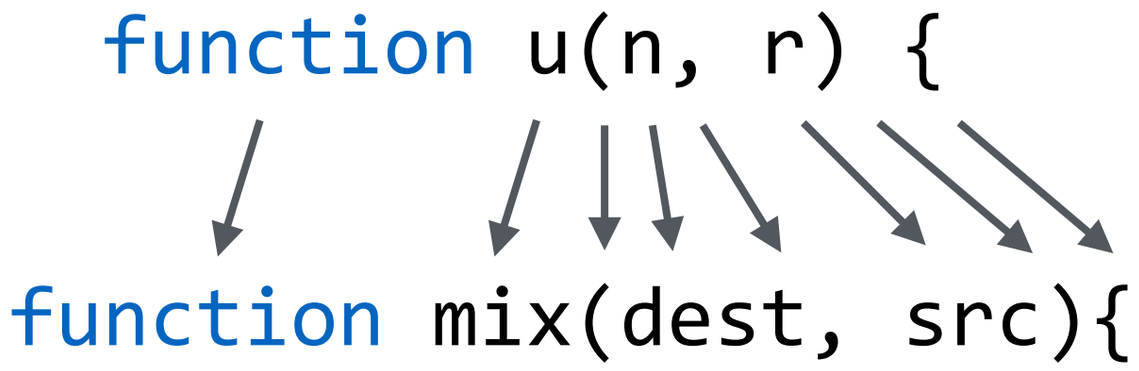
- Reordering
- Different length
- Dropped words

```
  function u(n, r) {


function mix(dest, src){
```

Alignment

Natural language:
non-trivial alignment

- Reordering
- Different length
- Dropped words



Minification:
straightforward
alignment

Complications

?

```
function r(n, r) {
  for (var t in r) n[t] = r[t];

  return n;

}
```

Complications

```
function r(n, r) {
  for (var t in r) n[t] = r[t];
  return n;
}
```

Complications

```
function r(n, r) {
  for (var t in r) n[t] = r[t];
  return n;
}
```

Complications


```
function r(n, r) {
  for (var t in r) n[t] = r[t];

  return n;

}
```

Complications

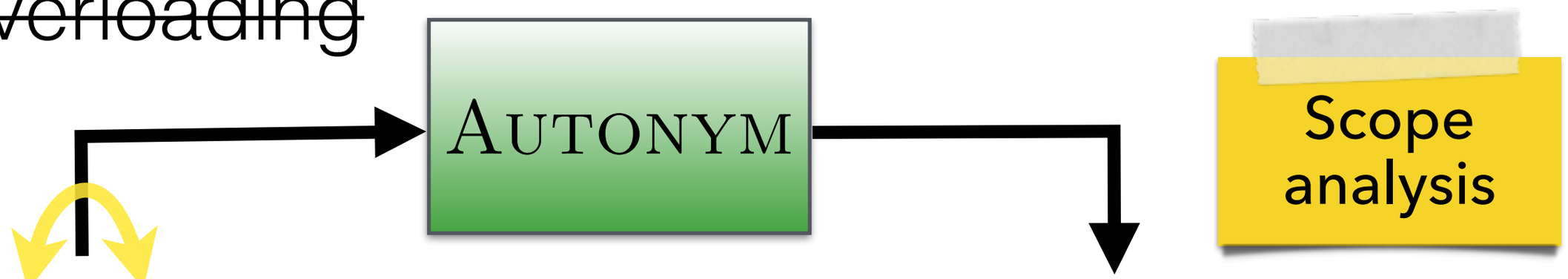
(1) Overloading


```
function r(n, r) {
  for (var t in r) n[t] = r[t];

  return n;

}
```

```
function mix(dest, src) {
}
```

Complications

~~(1) Overloading~~




```
function r(n, r) {
  for (var t in r) n[t] = r[t];

  return n;

}
```

```
function mix(dest, src) {
}
```

Complications

(2) Consistency

(Sentence-by-sentence translation)


```
function r(n, r) {
  for (var t in r) n[t] = r[t];

  return n;

}
```

```
function mix(dest, src) {
  for (var k in list) dest[k] = list[k];

  return dest;

}
```

Complications

~~(2) Consistency~~

(Sentence-by-sentence translation)

AUTONYM

Language model scoring

```
function r(n, r) {
  for (var t in r) n[t] = r[t];

  return n;

}
```

```
function mix(dest, src) {
  for (var k in list) dest[k] = list[k];

  return dest;

}
```

Idea: try all, let language model decide which is more natural, on average, across ALL lines

Evaluation

- Held-out test set: 2,149 files
- Comparison to JSNice [Raychev et al, 2015]
- Metric: % names recovered

Evaluation

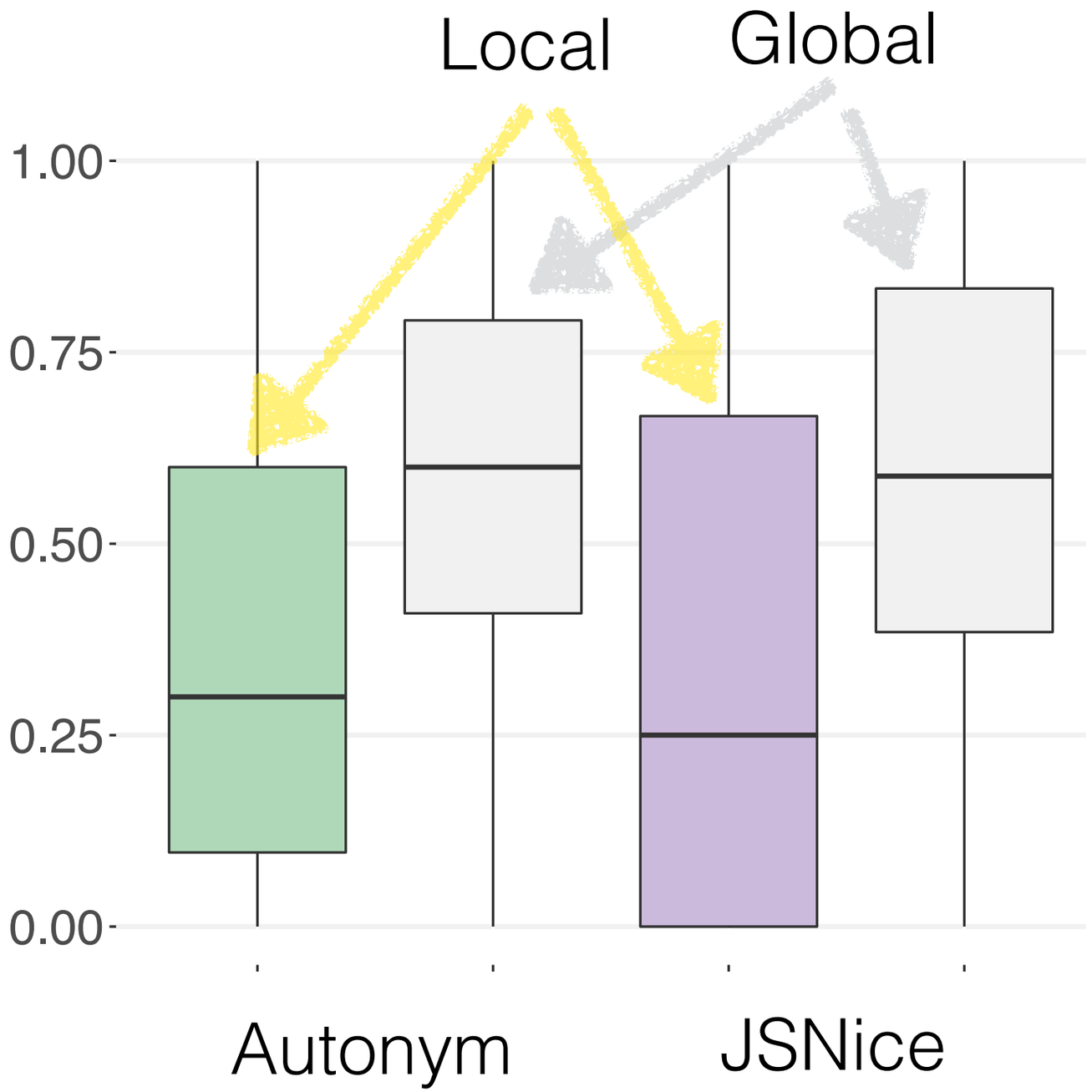
- Held-out test set: 2,149 files
- Comparison to JSNice [Raychev et al, 2015]
- Metric: % names recovered

  - Global vs. local names (globals don't change)

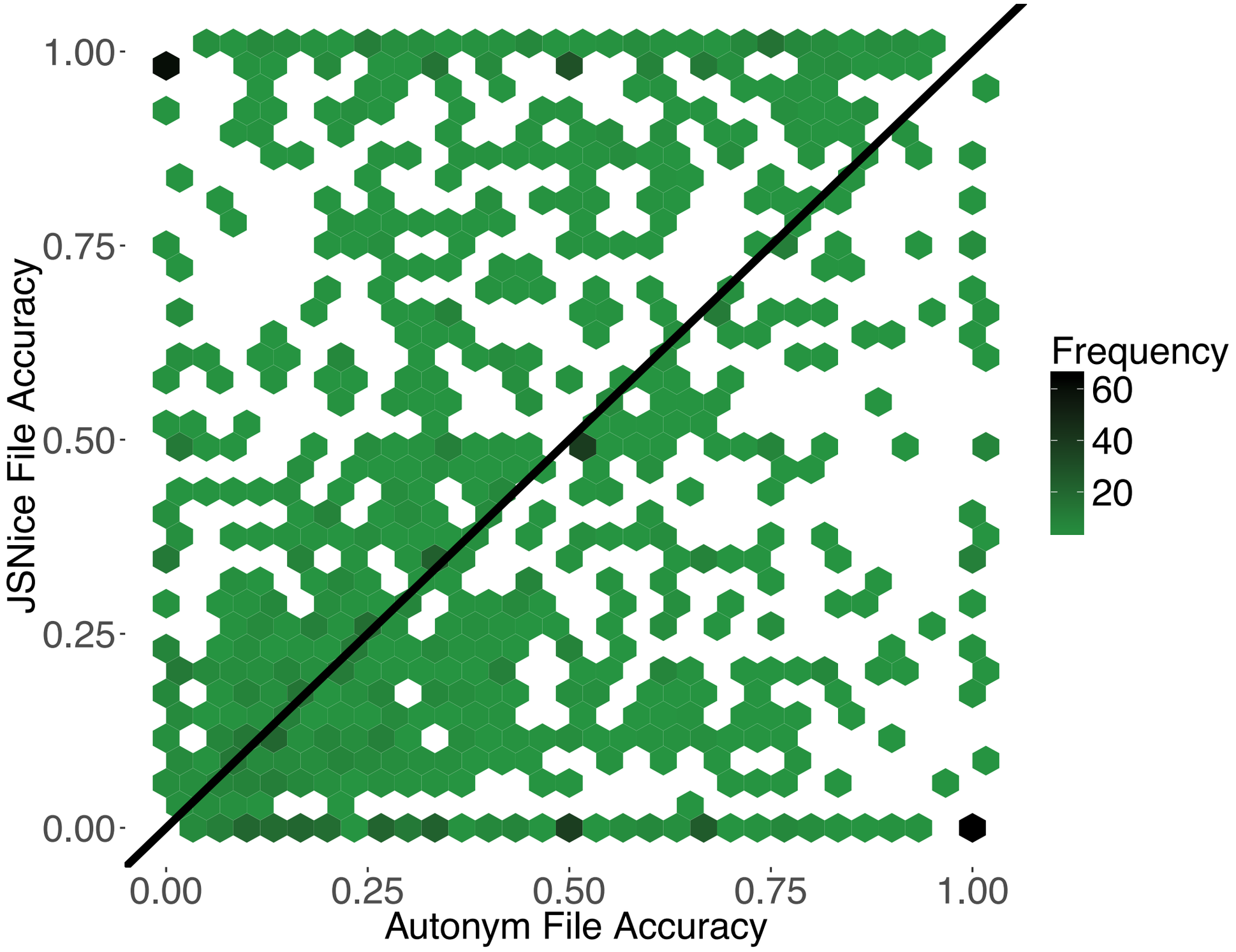
```
var geom2d = function() {
  var t = numeric.sum;
  function r(n, r) {
    this.x = n;
    this.y = r;
  }
...
```

```
var geom2d = function() {
  var sum = numeric.sum;
  function Vector2d(x, y) {
    this.x = x;
    this.y = y;
  }
...
```

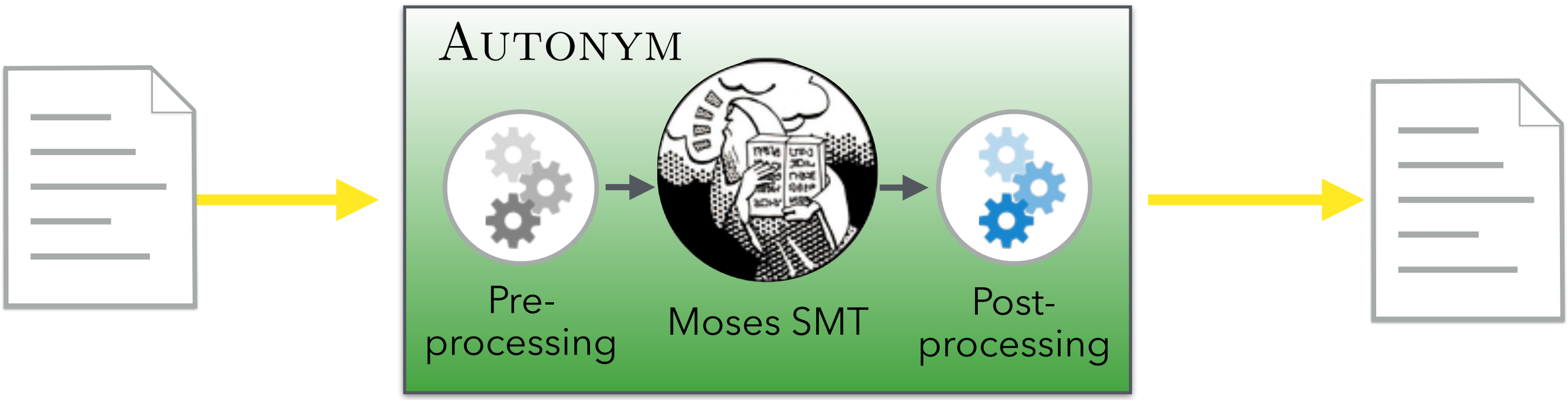
% names recovered (2,149 test files)



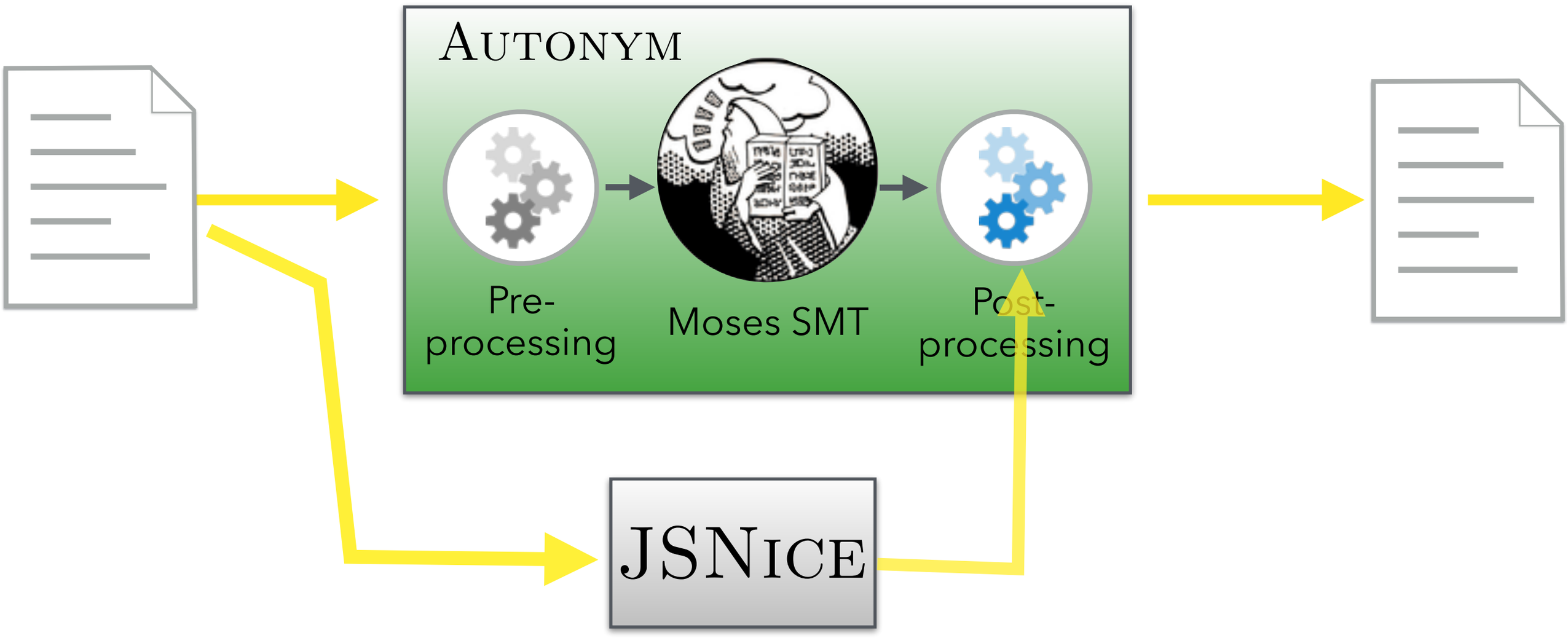
Joining forces



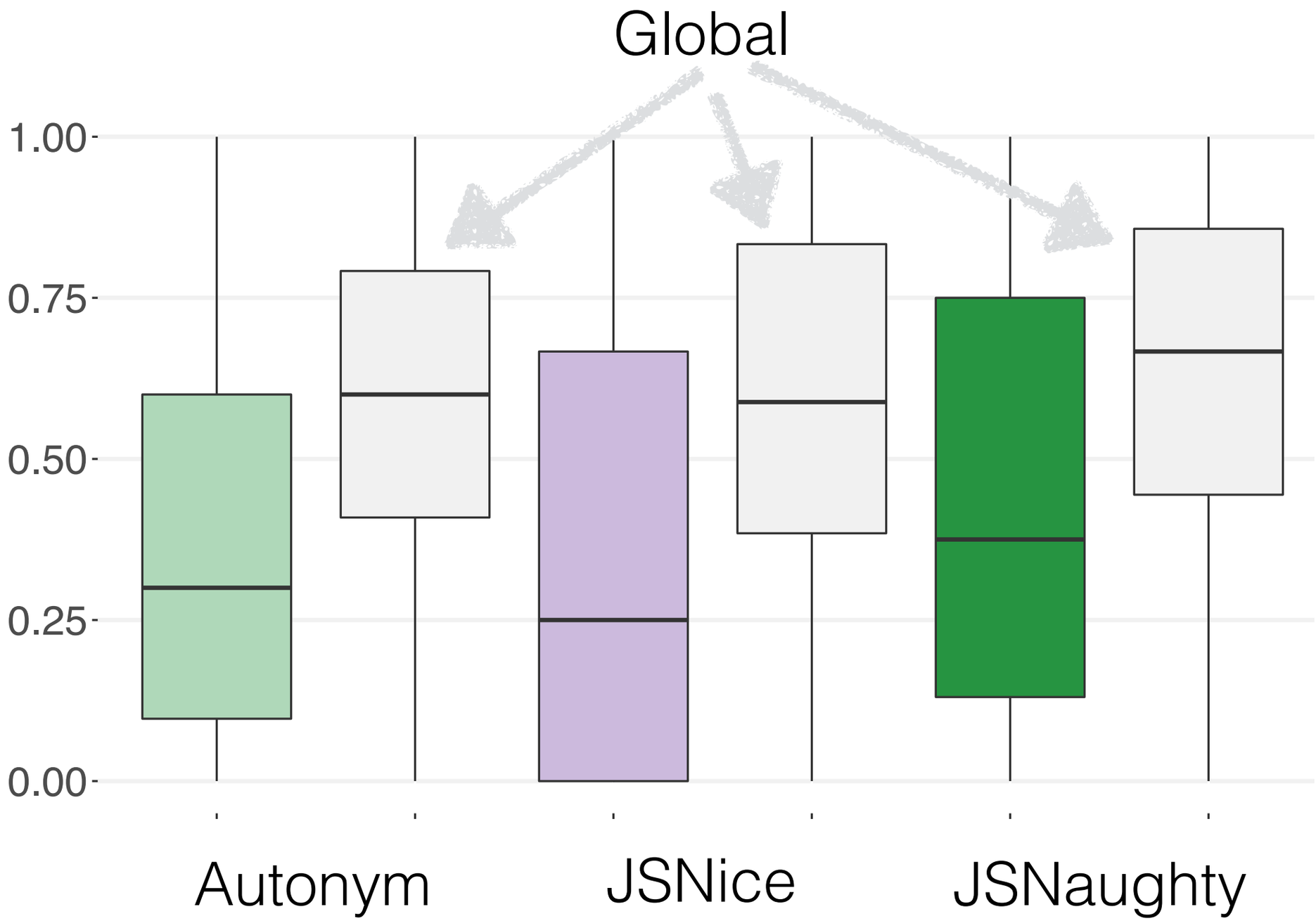
Becoming JSNaughty



Becoming JSNaughty



% names recovered (2,149 test files)



Examples

Original:	headers
AUTONYM	headers
JSNICE	headers
JSNAUGHTY	headers

Original:	jsonStream
AUTONYM	i
JSNICE	s
JSNAUGHTY	s

Original:	req
AUTONYM	req
JSNICE	q
JSNAUGHTY	req

Original:	res
AUTONYM	res
JSNICE	r
JSNAUGHTY	res

Original:	separator
AUTONYM	data
JSNICE	sep
JSNAUGHTY	sep

```
module.exports = http.createServer(function(e, r) {
  var t;
  var i = new stream.Stream();
  ...
  var n = "";
  csv().fromStream(e).on("data", function(e, r) {
    if (!t) {...}
    var a = {};
    _(_.zip(t, e)).each(function(e) {...});
    i.emit("data", n + JSON.stringify(a));
    n = ",";
  }).on("end", function(e) {
    i.emit("data", "]}");
    i.emit("end");
  }).on("error", function(e) {
    i.emit("error", e);
    console.log("csv error", e.mess
  });
});
```

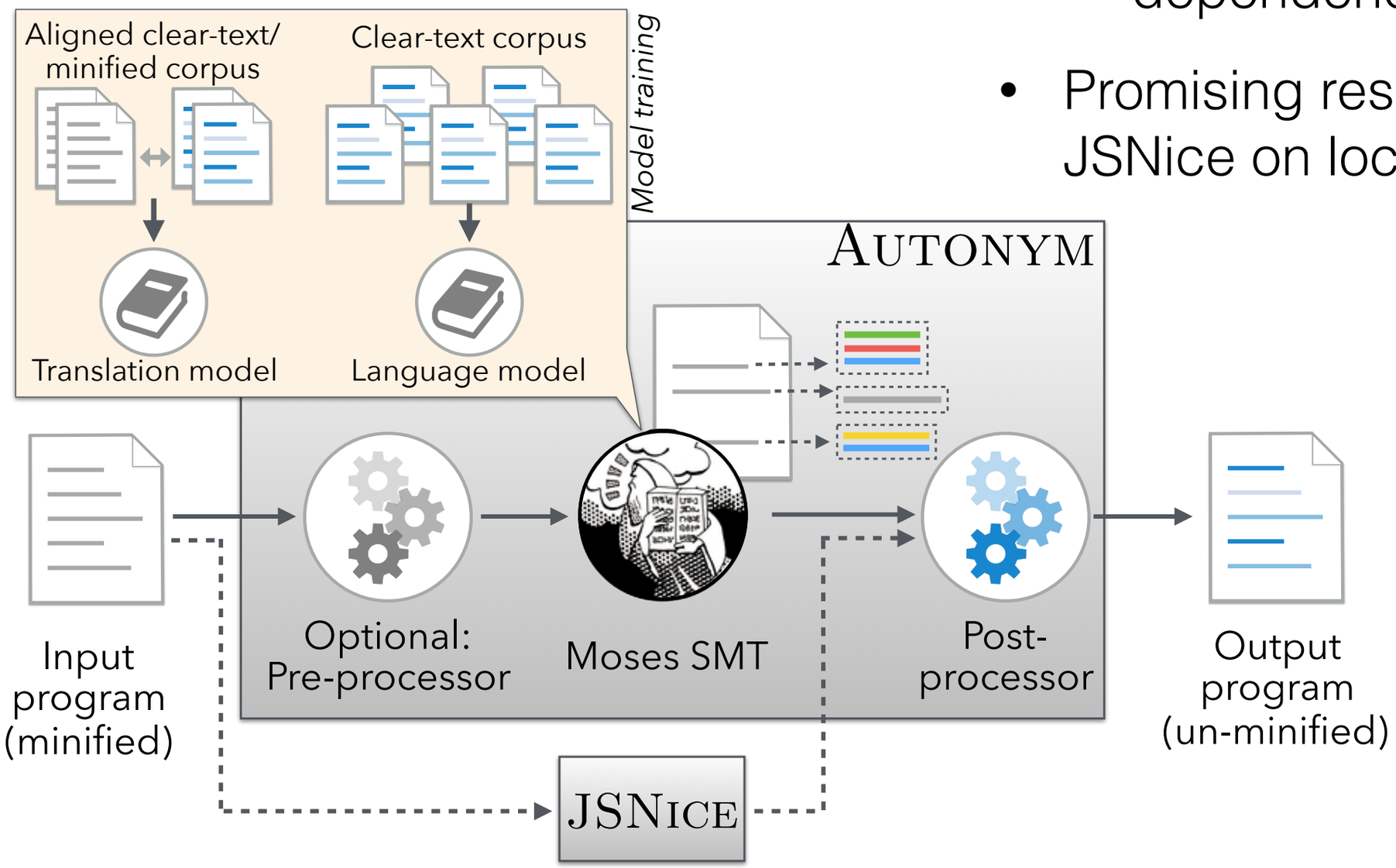
Original:	error
AUTONYM	err
JSNICE	err
JSNAUGHTY	err

Original:	tuple
AUTONYM	tuple
JSNICE	key
JSNAUGHTY	tuple

JSNaughty.org

https://github.com/bvasiles/jsNaughty

- Identifier renaming using SMT, e.g., minified JS, decompiled C
- Generic, mature off-the-shelf technology (Moses)
- Language dependence restricted to tokenization and scope analysis
  - dependency parse in JSNice
- Promising results: ~50% better than JSNice on local names, on average



This material is based upon work supported by the National Science Foundation under Grant No. 1414172

Machine translation for code

- Oda et al. (ASE ’15): *code to pseudocode*

```
# Python
if n % 3 == 0:
```

Pseudo-code:
if n is divisible by 3

- Karaivanov et al. (Onward! ’14): *porting C# to Java*

```
// C#
Console . WriteLine ( "Hello World!" ) ;
// Java
System . out . println ( "Hello World!" ) ;
```

Machine translation for code

- Oda et al. (ASE ’15):
  code to pseudocode

```
# Python
if n % 3 == 0:
```

Pseudo-code:
if n is divisible by 3

- Karaivanov et al. (Onward! ’14):
  porting C# to Java

```
// C#
Console . WriteLine ( "Hello World!" ) ;
// Java
System . out . println ( "Hello World!" ) ;
```

- Nguyen et al. (FSE’ 13, ASE ’15):
  porting Java to C#

```
// Java
public void findResultEdges() {
  for (Iterator it = dirEdgeList.iterator(); it.hasNext();) {
    DirectedEdge de = (DirectedEdge) it.next();…}
}

// C#
public void FindResultEdges() {
  foreach (DirectedEdge de in _dirEdgeList){…}
}
```