PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

Learning to Map Natural Language to General Purpose Source Code

Tasks | Resources | Models

Srini Iyer

Joint work with:
Luke Zettlemoyer, Alvin Cheung, Yannis Konstas, Jayant Krishnamurthy

NL→General Source Code

Natural Language Intents

Add a scalar to this vector in place.

```java
public void add(final double arg0) {

    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }

  }
```

Java

Plot data as a scatterplot with double-lined hexagons.

```python
plt.scatter(x, y, s=400, c=whites, marker='h', alpha=0.5,
edgecolors='black', linewidth=1)

plt.scatter(x, y, s=260, c=colors, marker='h', alpha=0.5,
edgecolors='black', linewidth=1)

plt.show()
```

Python

General Purpose Languages

Multiple Applications

Unrestricted Structure

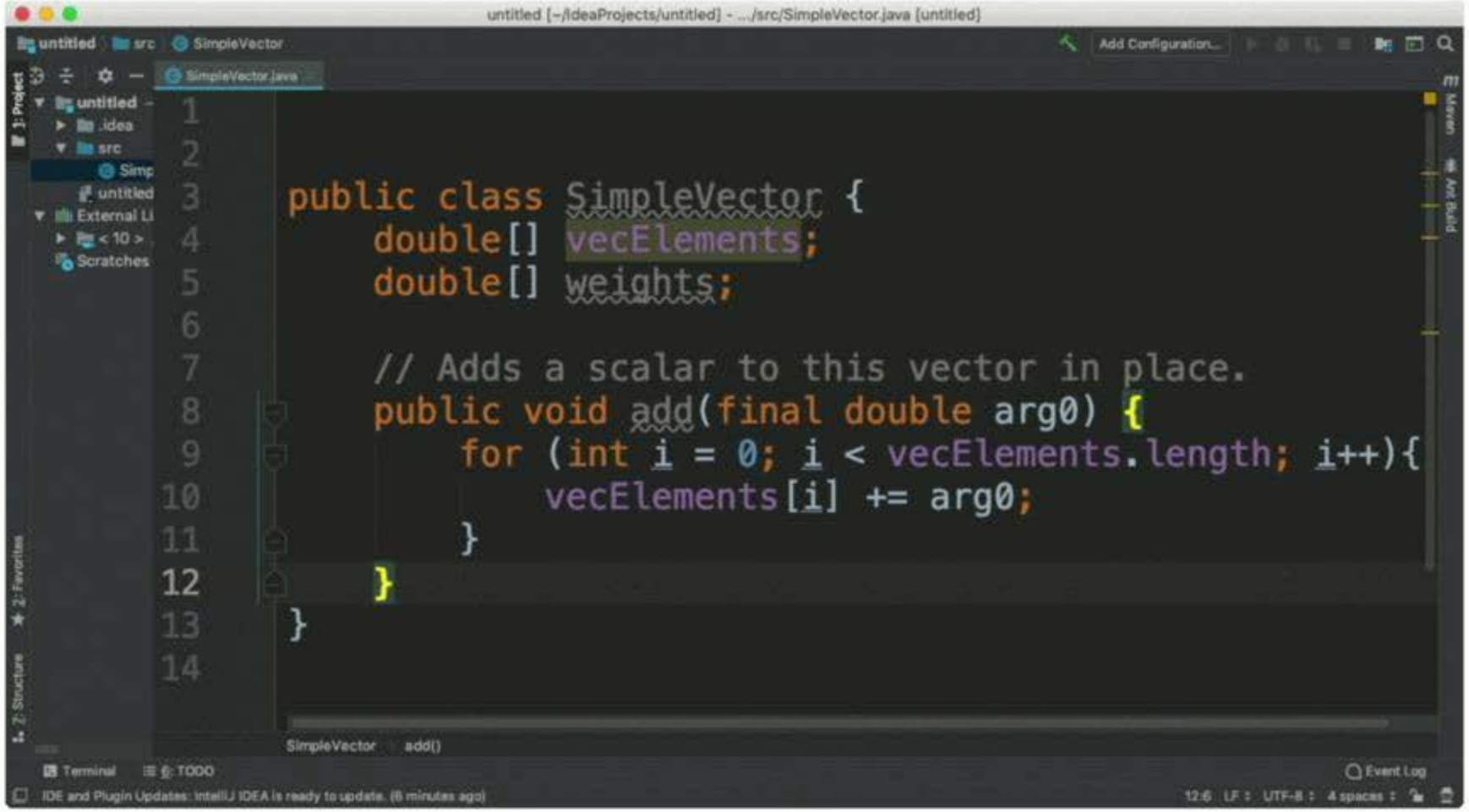
Inexpensive Supervision

Two Primary Audiences

Developer

Non-Expert User

NL Assistant in Programming IDEs



↑ Speed ↑ Efficiency ↑ Readable

Checking code Efficiency/Consistency

```
public class Violations {
  long numPptEntries;
  List<Violation> violations;









}
```

Checking code Efficiency/Consistency

```
public class Violations {
  long numPptEntries;
  List<Violation> violations;

  // Empty the violations list.




}
```

Checking code Efficiency/Consistency

```
public class Violations {
  long numPptEntries;
  List<Violation> violations;

  // Empty the violations list.
   public void empty() {
        violations = new Arraylist<Violation>();
     }
}
```

Checking code Efficiency/Consistency

```
public class Violations {
  long numPptEntries;
  List<Violation> violations;

  // Empty the violations list.
   public void empty() {
        violations = new Arraylist<Violation>();
     }
}
```

It looks like you're trying to empty a list.

I might have a more efficient way to do this:

```
void function() {
  violations.clear();
}
```

How to do X in language Y ?

How do I use reflection to call a generic method in C#?

How to do X in language Y ?

How do I use reflection to call a generic method in C#?

```
MethodInfo method = typeof(Sample).GetMethod("GenericMethod");
MethodInfo generic = method.MakeGenericMethod(myType);
generic.Invoke(this, null);
```

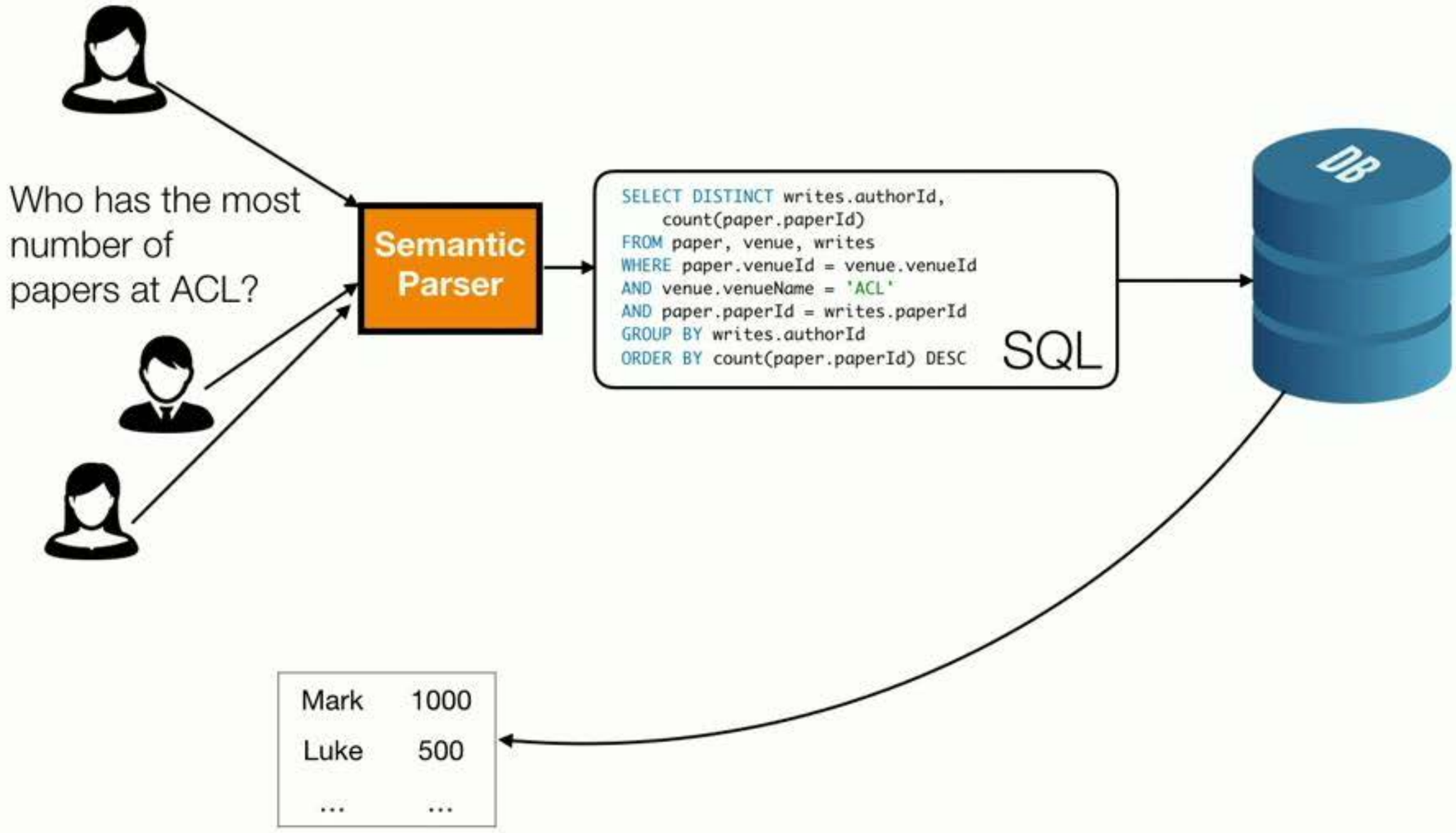
How to do X in language Y ?

How do I use reflection to call a generic method in C#?

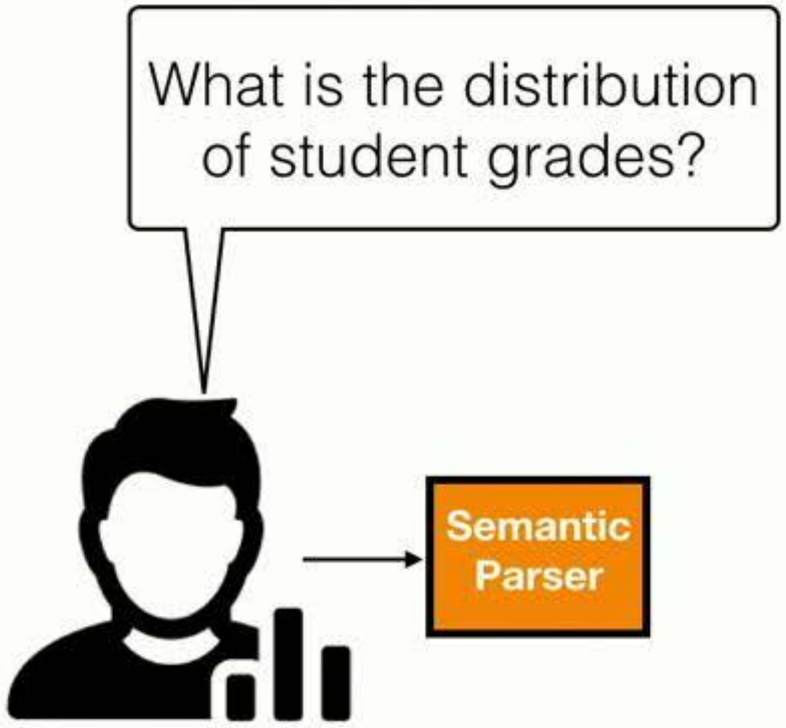
```
MethodInfo method = typeof(Sample).GetMethod("GenericMethod");
MethodInfo generic = method.MakeGenericMethod(myType);
generic.Invoke(this, null);
```

Stackoverflow Keyboard

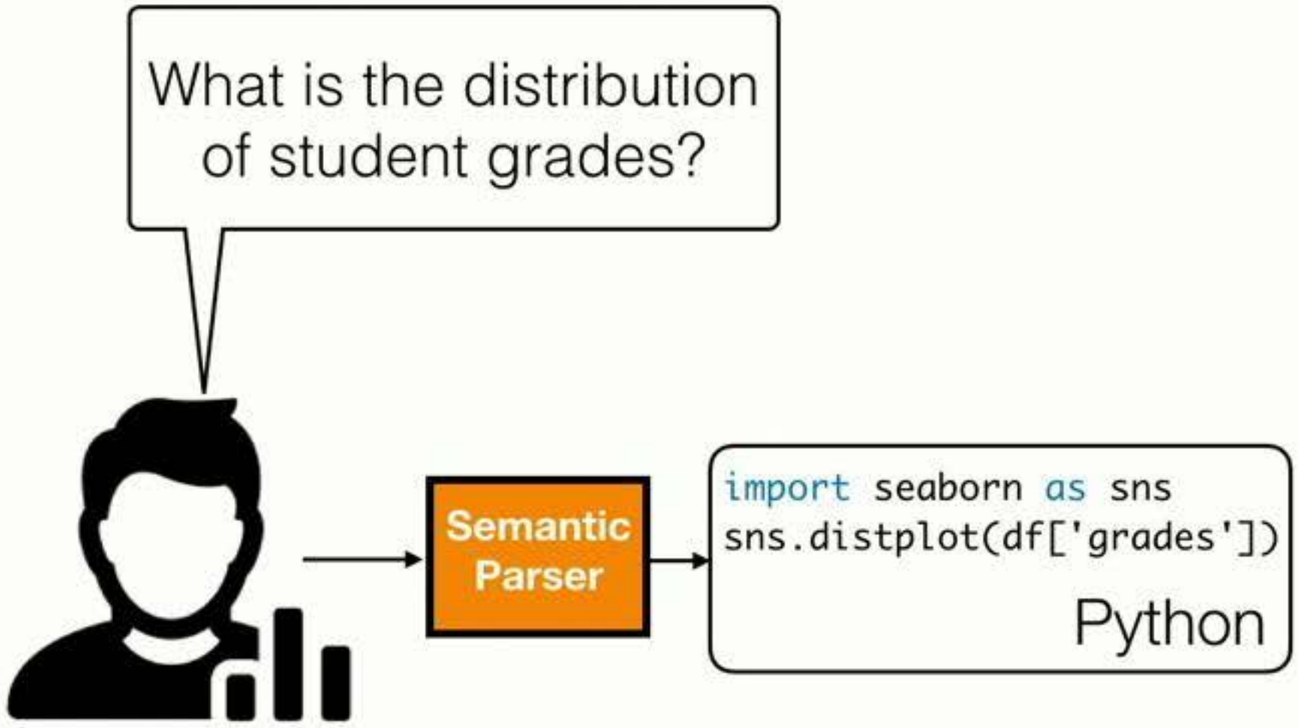
Querying Databases



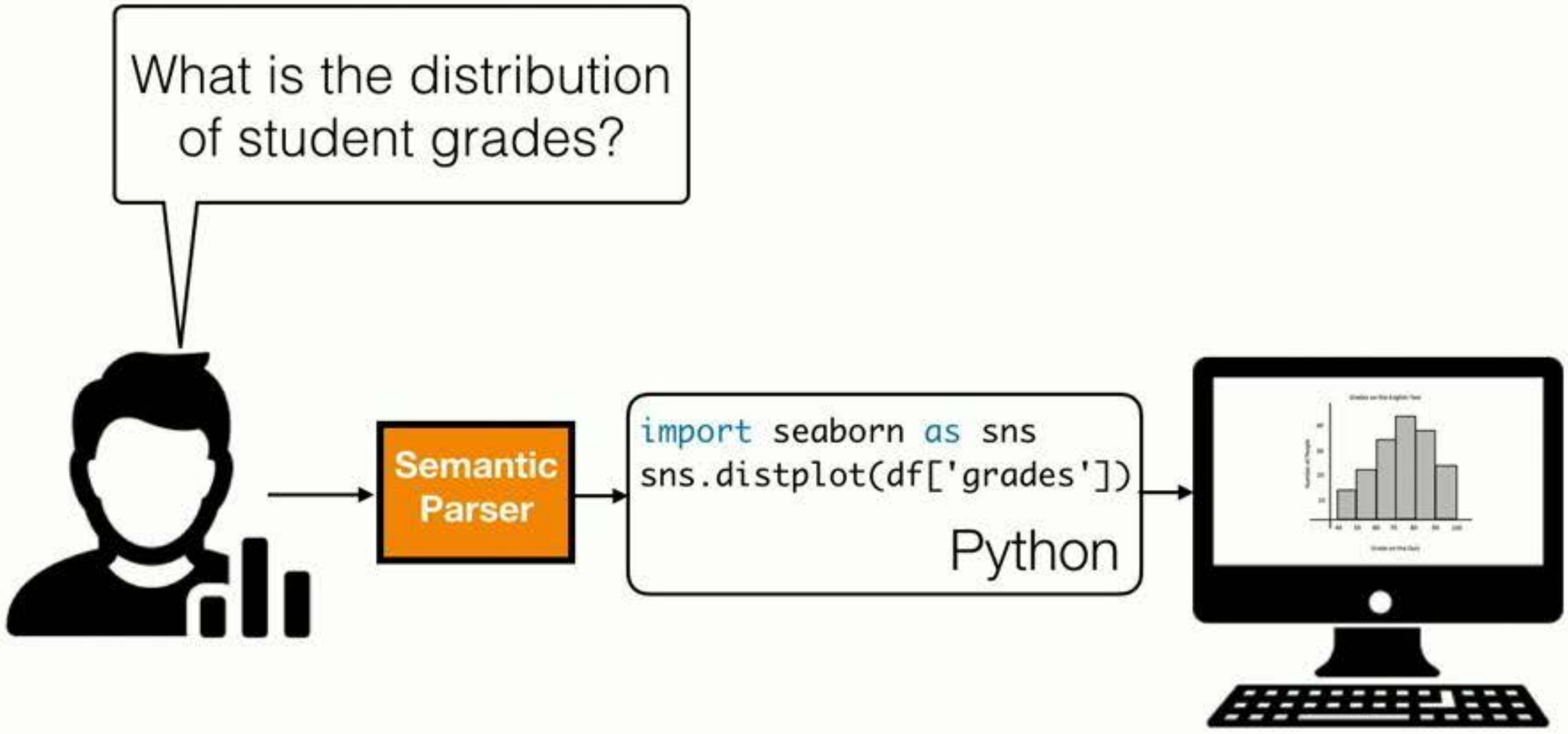
Visualizing Data



Visualizing Data



Visualizing Data



Two Audiences

Developer

Non-Expert User

Two Audiences

Developer

Non-Expert User

1. Code is visible

Two Audiences

Developer

1. Code is visible

Non-Expert User

1. Code is latent

Two Audiences

Developer

1. Code is visible
2. Partially correct code is also useful

Non-Expert User

1. Code is latent

Two Audiences

Developer

1. Code is visible
2. Partially correct code is also useful

Non-Expert User

1. Code is latent
2. Code should be exactly right

Two Audiences

Developer

1. Code is visible
2. Partially correct code is also useful

Non-Expert User

1. Code is latent
2. Code should be exactly right
3. Explanation is essential for trust

Two Audiences

Developer

1. Code is visible
2. Partially correct code is also useful
3. Explanation serves as documentation

Non-Expert User

1. Code is latent
2. Code should be exactly right
3. Explanation is essential for trust

Talk Outline

1. Mapping NL to Source Code in Programmatic Context

Srini Iyer, Yannis Konstas, Alvin Cheung, Luke Zettlemoyer,
Mapping Language to Code in Programmatic Context, EMNLP 2018

Srini Iyer, Alvin Cheung, Luke Zettlemoyer,
Learning Programmatic Idioms for Scalable Semantic Parsing, EMNLP 2019

2. Learning a Extensible Semantic Parser using User Feedback

Srini Iyer, Yannis Konstas, Alvin Cheung, Jayant K., Luke Zettlemoyer,
Learning a Neural Semantic Parser from User Feedback, ACL 2017

3. Topics for future research

Alane Suhr, **Srini Iyer,** Yoav Artzi [Outstanding Paper]
Learning to Map Context Dependent Sentences to Executable Formal Queries, NAACL 2018

Rajas Agashe, **Srini Iyer,** Luke Zettlemoyer
JulCEe: A Large Scale Distantly Supervised Dataset for Open Domain Context-based Code , EMNLP 2019

Talk Outline

1. Mapping NL to Source Code in Programmatic Context

   Srini Iyer, Yannis Konstas, Alvin Cheung, Luke Zettlemoyer,
   Mapping Language to Code in Programmatic Context, EMNLP 2018

   Srini Iyer, Alvin Cheung, Luke Zettlemoyer,
   Learning Programmatic Idioms for Scalable Semantic Parsing, EMNLP 2019

2. Learning a Extensible Semantic Parser using User Feedback

   Srini Iyer, Yannis Konstas, Alvin Cheung, Jayant K., Luke Zettlemoyer,
   Learning a Neural Semantic Parser from User Feedback, ACL 2017

3. Topics for future research

   Alane Suhr, **Srini Iyer,** Yoav Artzi [Outstanding Paper]
   Learning to Map Context Dependent Sentences to Executable Formal Queries, NAACL 2018

   Rajas Agashe, **Srini Iyer,** Luke Zettlemoyer
   JuICEe: A Large Scale Distantly Supervised Dataset for Open Domain Context-based Code , EMNLP 2019

Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;



}
```

Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;


}
```

Context

Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

  NL Query: Adds a scalar to this vector in place.

}
```

Context

Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

  NL Query: Adds a scalar to this vector in place.
  public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }


}
```

Context

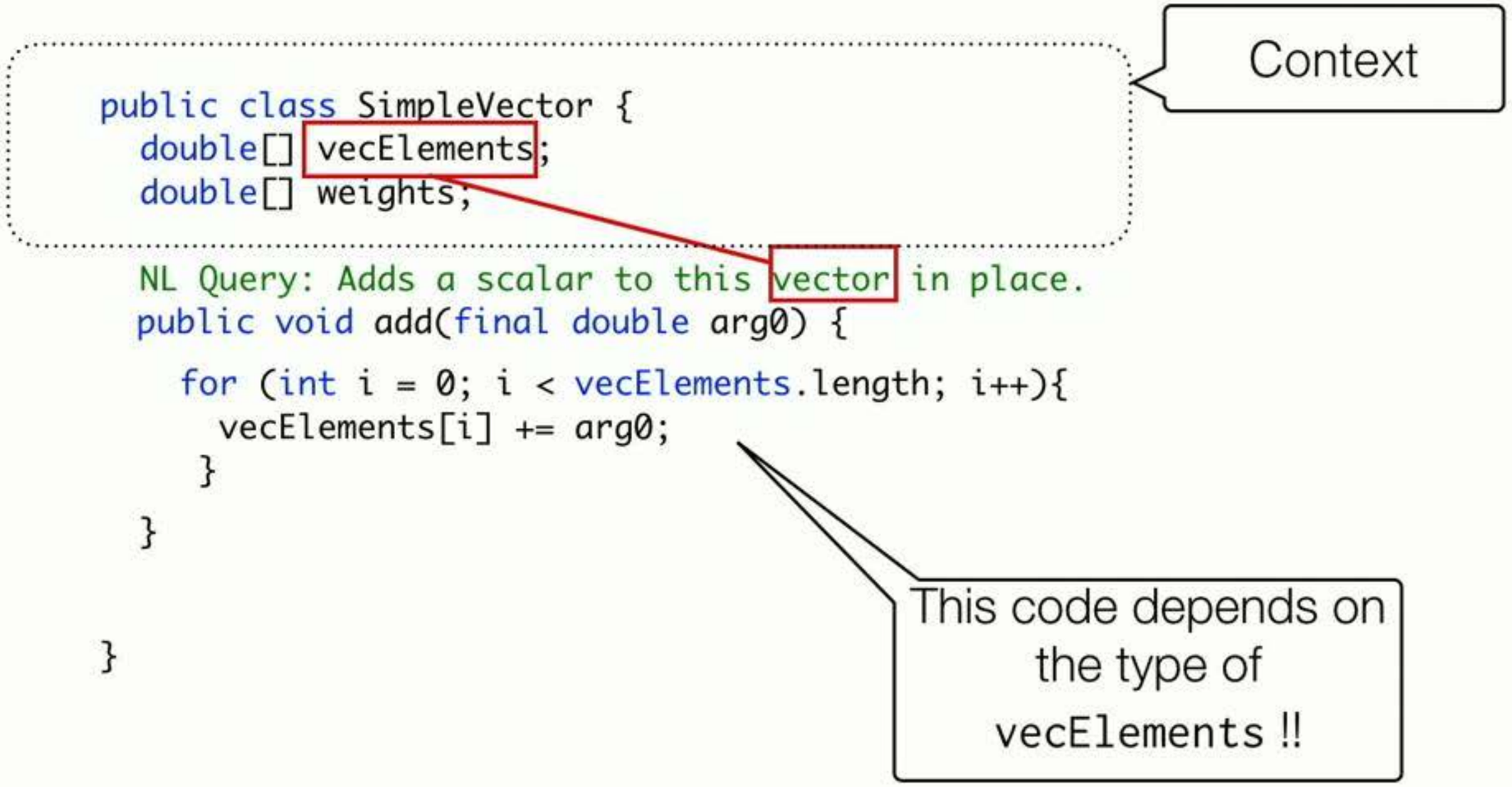
Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

  NL Query: Adds a scalar to this vector in place.
  public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }


}
```

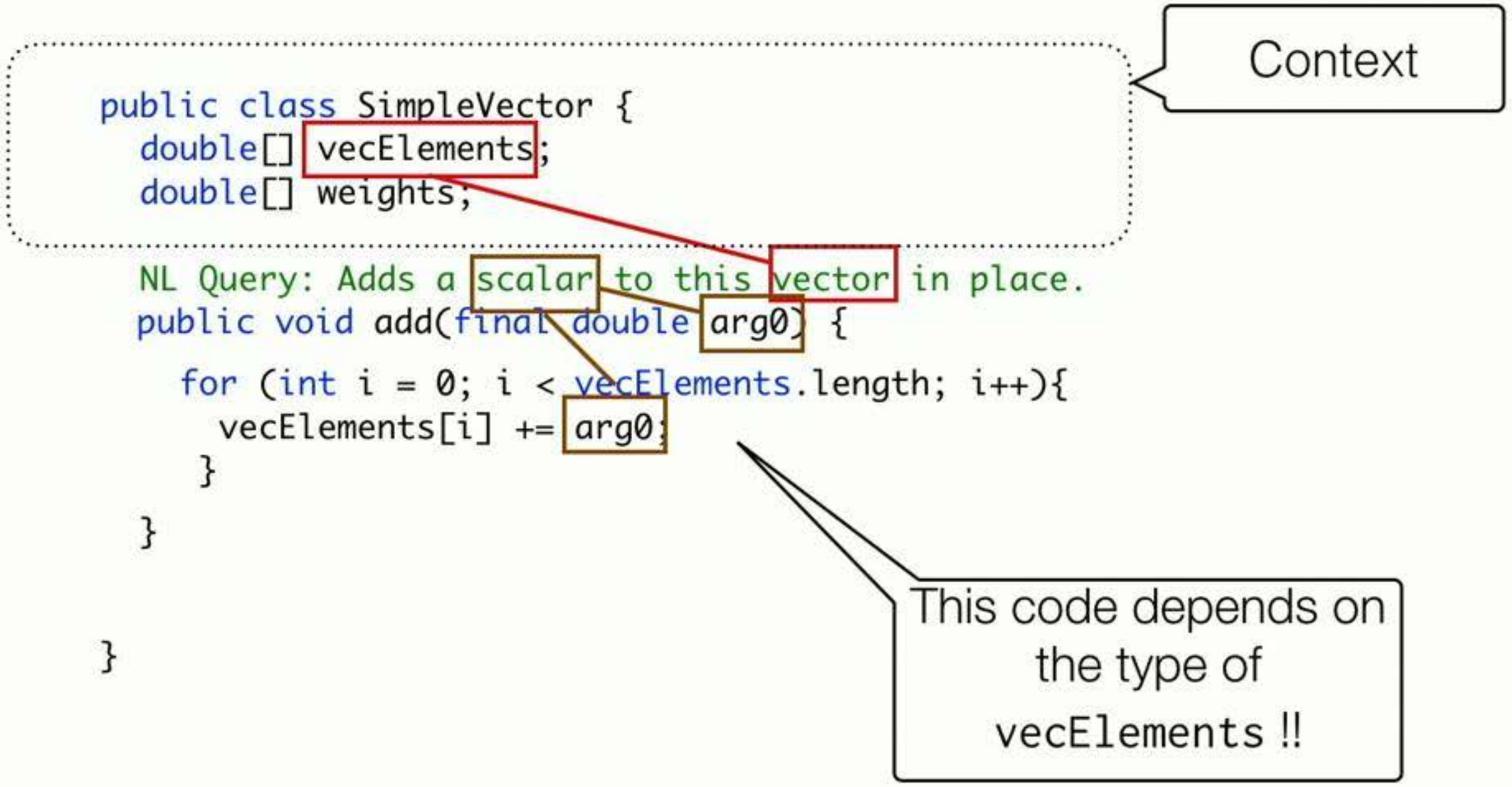
Context

This code depends on the type of `vecElements` !!

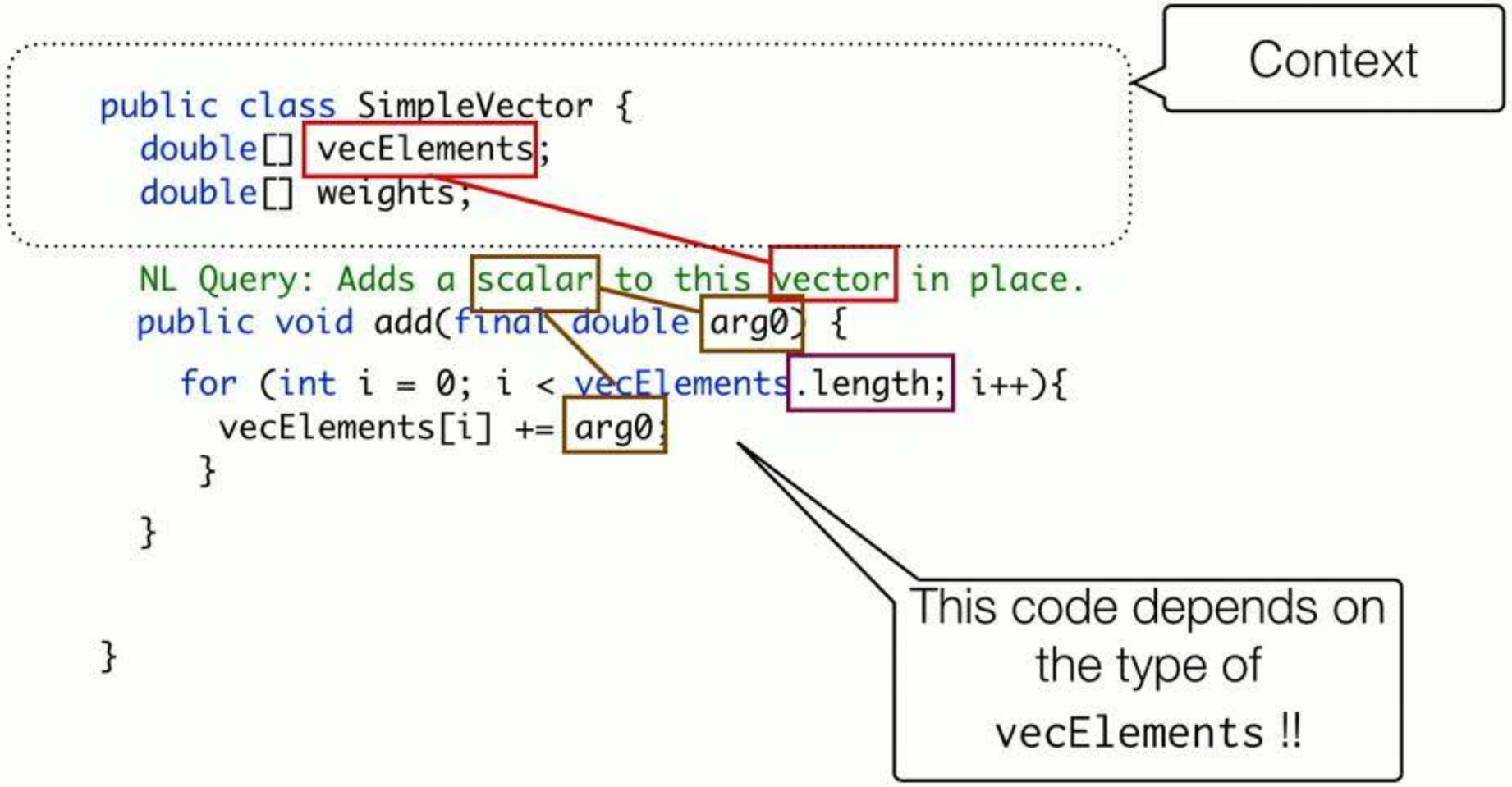
Contextual Code Generation



Contextual Code Generation



Contextual Code Generation



Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

NL Query: Adds a scalar to this vector in place.
  public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }

NL Query: Increment this vector

}
```

Context

Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

NL Query: Adds a scalar to this vector in place.
  public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }

NL Query: Increment this vector
  public void inc() {
    this.add(1);
  }
}
```

Context

Contextual Code Generation

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

NL Query: Adds a scalar to this vector in place.
  public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }

NL Query: Increment this vector
  public void inc() {
    this.add(1);
  }
}
```

Context

This code depends on the existence of `add()`.

CONCODE Dataset

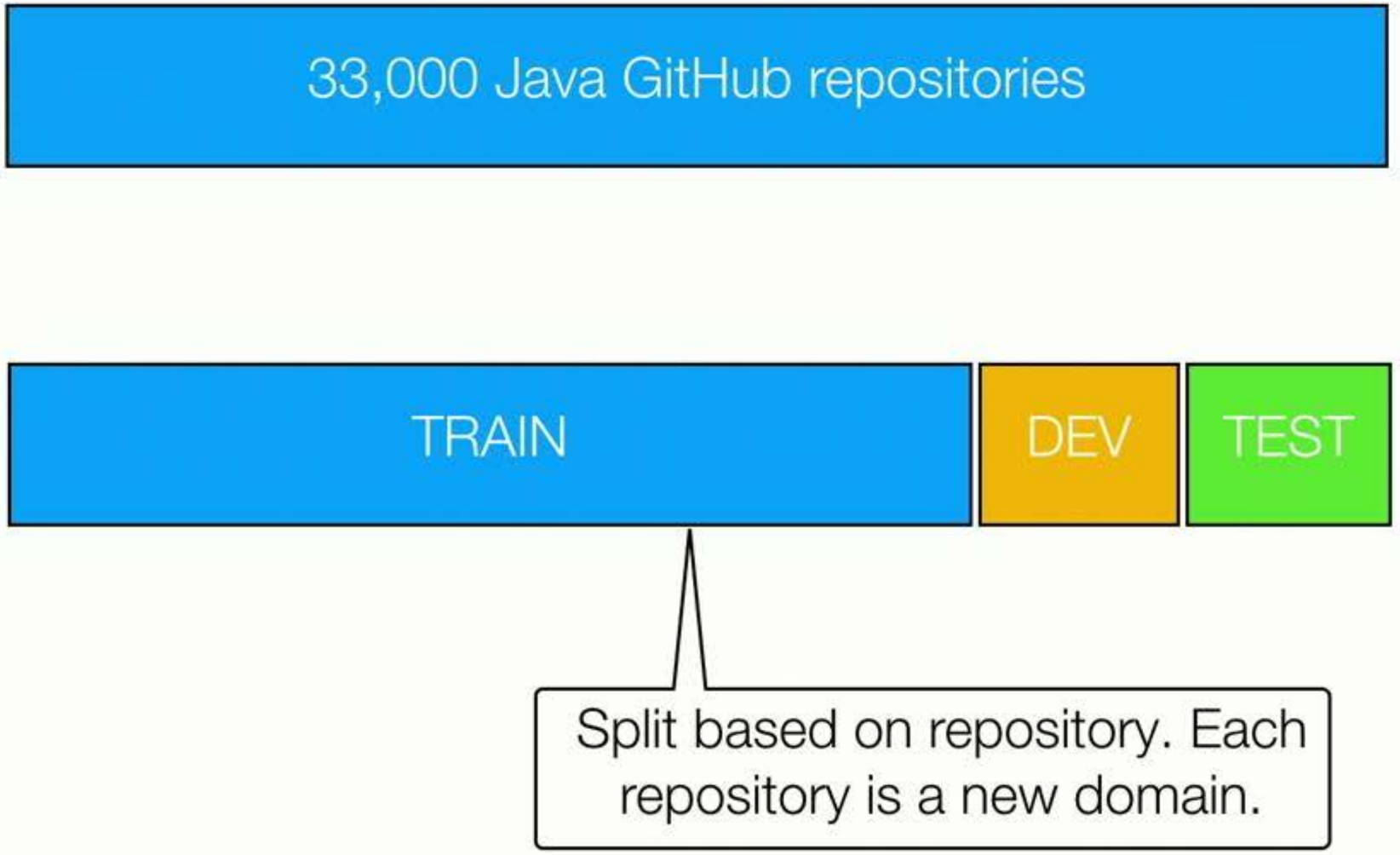
CONCODE Dataset

CONCODE Dataset

33,000 Java GitHub repositories

Learn from inexpensive data from the web

CONCODE Dataset



CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /**
     *  Adds a scalar to this vector in place.
     *  @param num
     */
    public void add(final double num) {
    for (int i = 0; i < vecElements.length; i++){
       vecElements[i] += num;
     }
  }

    /**
     *  Increment this vector
     */
    public void inc() {
     this.add(1);
  }
}
```

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /**
     *  Adds a scalar to this vector in place.
     *  @param num
     */
    public void add(final double num) {
    for (int i = 0; i < vecElements.length; i++){
       vecElements[i] += num;
     }
  }

    /**
     *  Increment this vector
     */
    public void inc() {
     this.add(1);
  }
}
```

Remove derived classes

Remove special tags

CONCODE Dataset

Context + NL

Target Code

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /*  Adds a scalar to this vector in place. */

    public void add(final double num) {
    for (int i = 0; i < vecElements.length; i++){
       vecElements[i] += num;
     }
  }

    /* Increment this vector */
    public void inc() {
     this.add(1);
  }
}
```

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /*  Adds a scalar to this vector in place. */

    public void add(final double num) {
    for (int i = 0; i < vecElements.length; i++){
       vecElements[i] += num;
     }
  }

    /* Increment this vector */
    public void inc() {
     this.add(1);
  }
}
```

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */
    public void add (final double num) {
      for (int i = 0; i < vecElements.length; i++) {
        vecElements[loc0] += num;
      }
  }
    /* Increment this vector */
    public void inc() {
     this.add(1);
  }
}
```

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */
    public void function_name (final double num) {
      for (int i = 0; i < vecElements.length; i++) {
        vecElements[loc0] += num;
      }
  }

    /* Increment this vector */
    public void inc() {
     this.add(1);
  }
}
```

Replace method name

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */
    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    /* Increment this vector */
    public void inc() {
     this.add(1);
  }
}
```

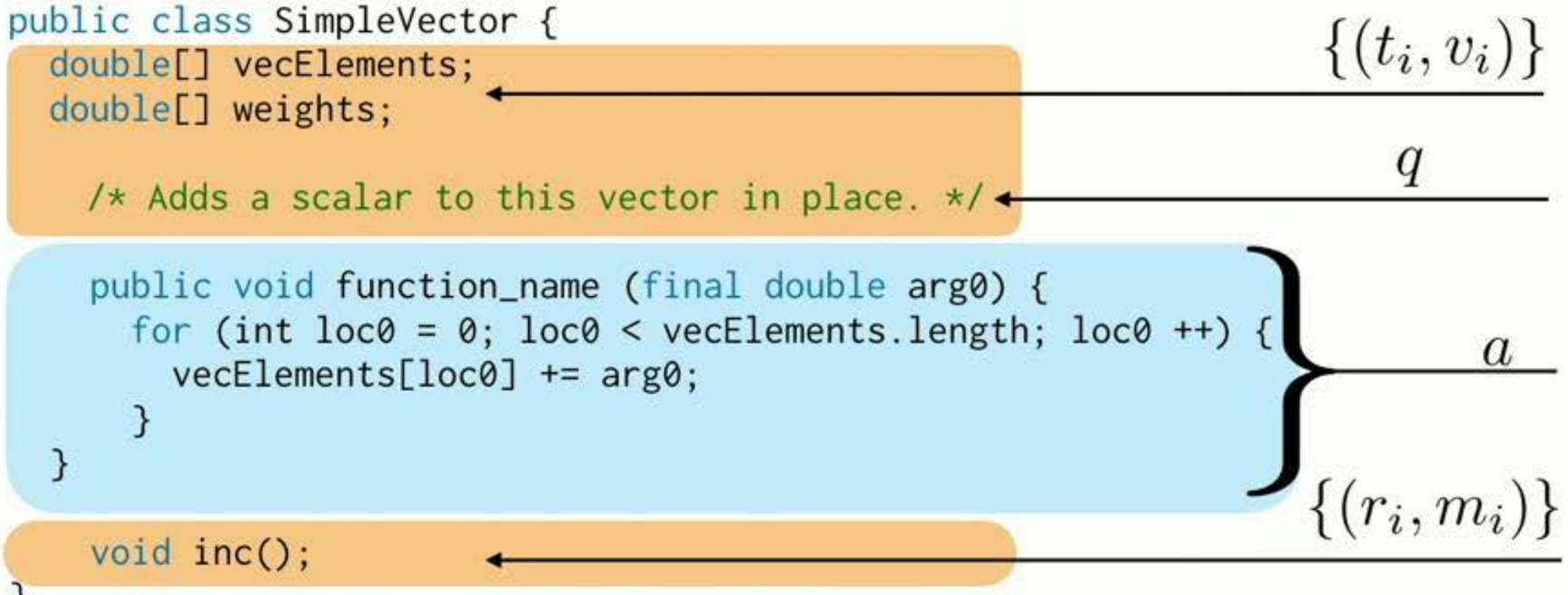
Rename parameters and local variables

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Strip params and body of Context Methods

CONCODE Dataset



Context + NL

Target Code

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Context + NL

Target Code

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Avg: 5 variables

Context + NL

Target Code

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Avg: 5 variables

13.7 words on average

Context + NL

Target Code

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Avg: 5 variables

13.7 words on average

26.3 tokens on average

Context + NL

Target Code

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Avg: 5 variables

13.7 words on average

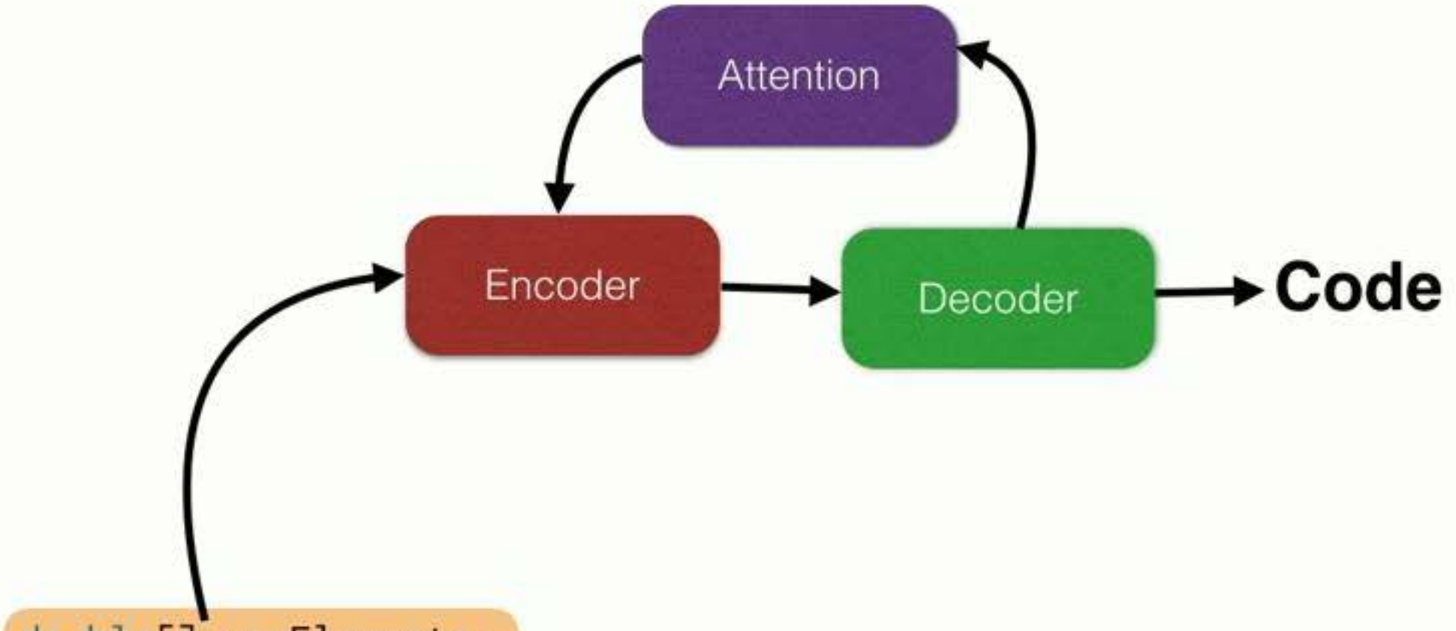
26.3 tokens on average

Avg: 11 methods

Context + NL

Target Code

Encoder-Attention-Decoder




```
double[] vecElements;
double[] weights;

public void inc() ;

Adds a scalar to this
vector in place.
```

CONCODE Dataset

```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Context + NL

Target Code

CONCODE Dataset

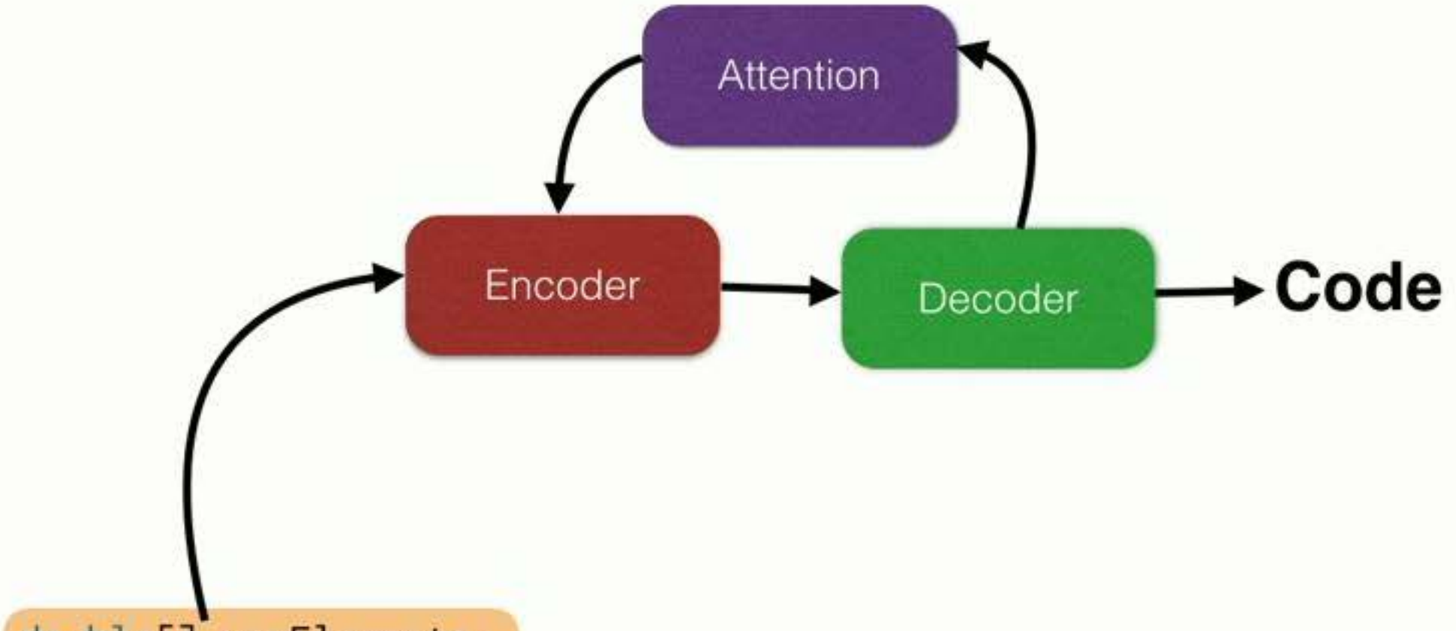
```
public class SimpleVector {
  double[] vecElements;
  double[] weights;

    /* Adds a scalar to this vector in place. */

    public void function_name (final double arg0) {
      for (int loc0 = 0; loc0 < vecElements.length; loc0 ++) {
        vecElements[loc0] += arg0;
      }
  }

    void inc();
}
```

Avg: 5 variables

Context + NL

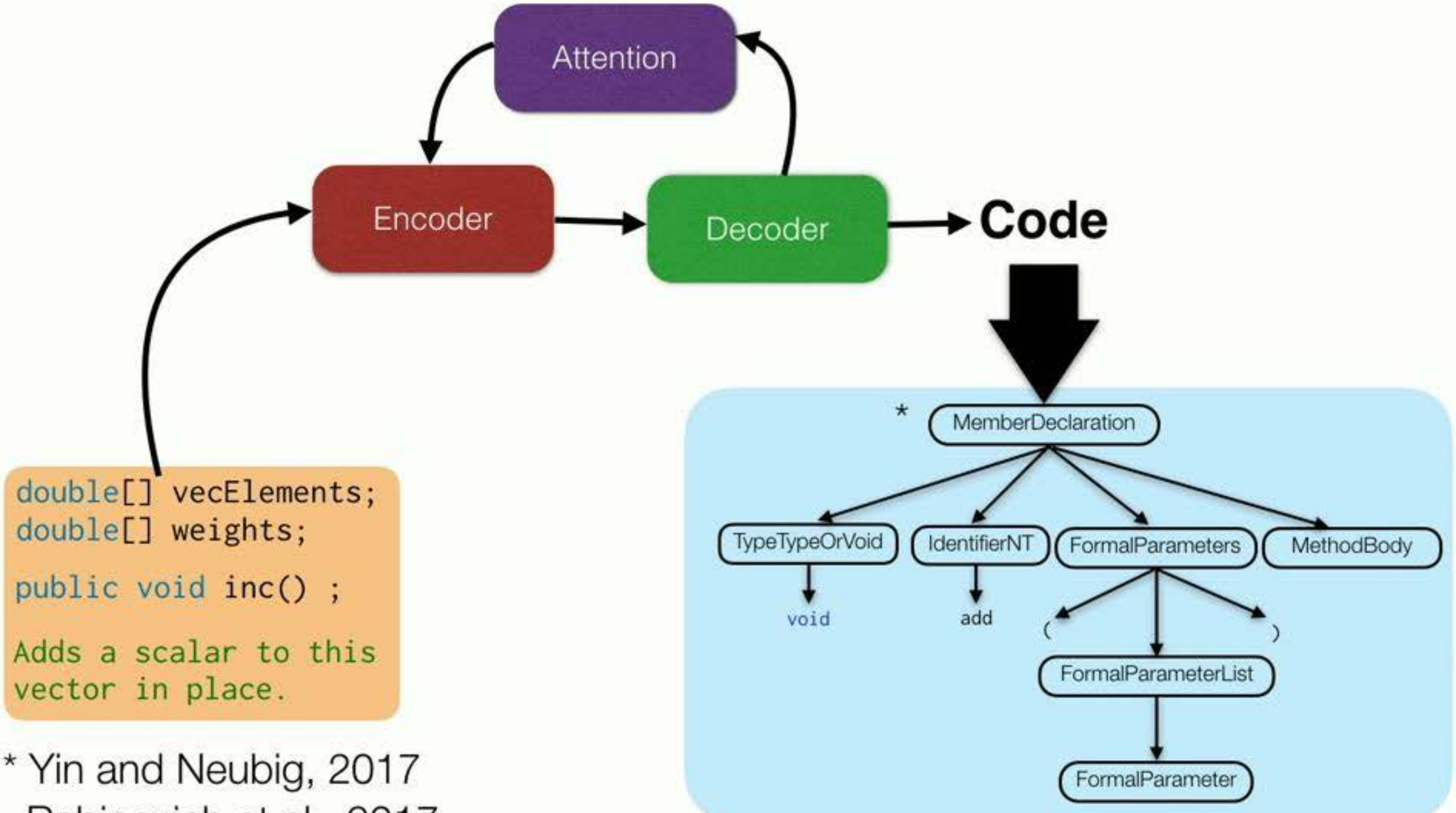
Target Code

Encoder-Attention-Decoder




```
double[] vecElements;
double[] weights;

public void inc() ;

Adds a scalar to this
vector in place.
```

Encoder-Attention-Decoder



* Yin and Neubig, 2017
  Rabinovich et al., 2017

Encoding the Environment

```
double[] vecElements;
double[] weights;

void inc() ;

Adds a scalar to this
vector in place.
```

Variables

Methods

NL Query

Encoding tokens

```
MyReally_ComplexType<Integer>[]
```

Encoding tokens

Separate special symbols

```
MyReally_ComplexType < Integer > [ ]
```

Encoding tokens

Split based on underscores

```
MyReally ComplexType < Integer > [ ]
```

Encoding tokens

Split based on Camel Casing

```
My Really Complex Type < Integer > [ ]
```

Encoding tokens

Lowercase tokens

```
my really complex type < integer > [ ]
```

Encoding tokens

Byte pair encoding (Sennrich et al. 2015)

```
my real ly comp lex type < int eger > [ ]
```

Encoding the Environment

```
double[] vecElements;
double[] weights;

void inc() ;

Adds a scalar to this
vector in place.
```

Variables

Methods

NL Query

Encoding the Environment

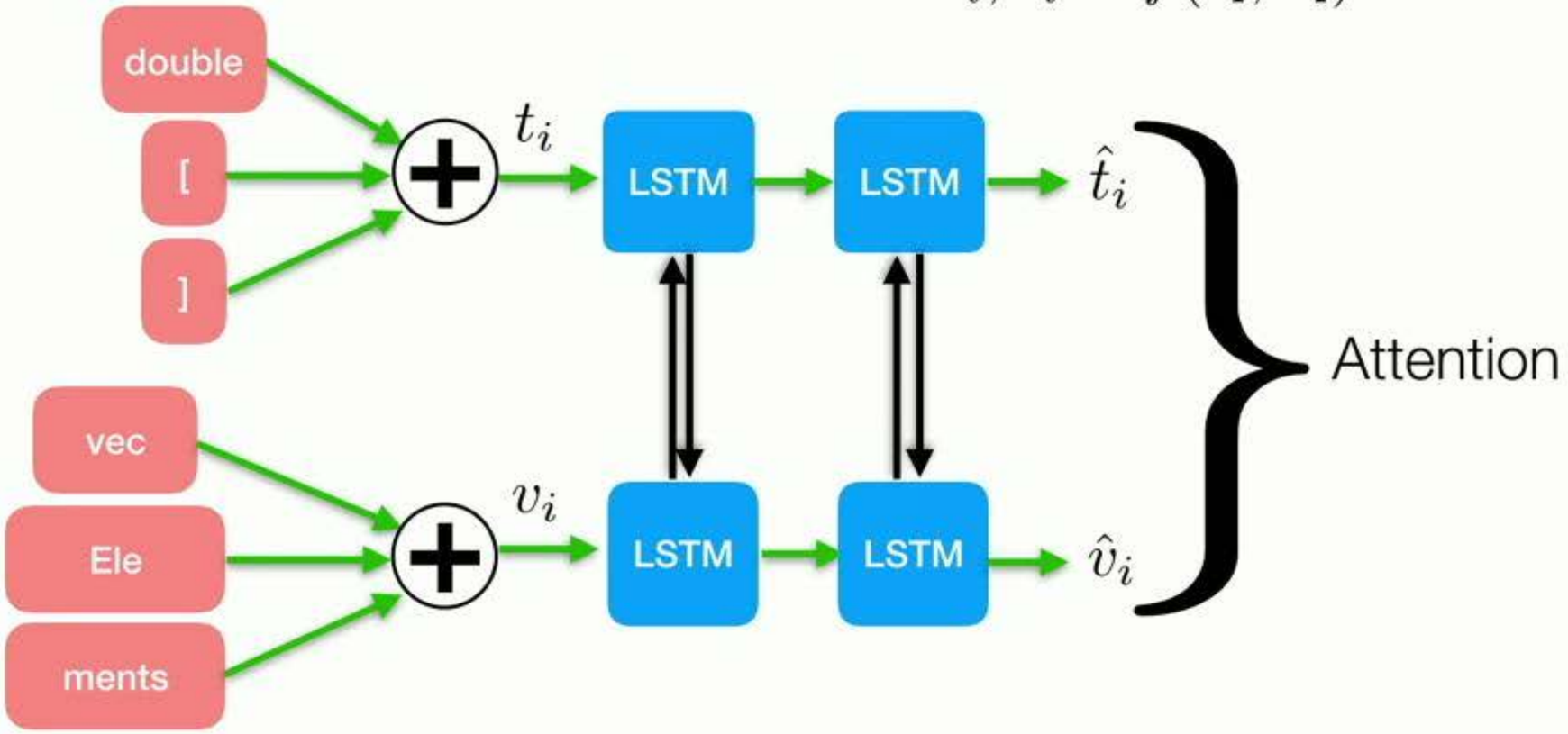
double [ ]

vec Ele ments

$$\mathbf{t_i} = Avg(\mathbf{t_{i1}}, \ldots, \mathbf{t_{ij}})$$

$$\mathbf{v_i} = Avg(\mathbf{v_{i1}}, \ldots, \mathbf{v_{ij}})$$

$$\hat{t}_i, \hat{v}_i = f(\mathbf{t_i}, \mathbf{v_i})$$

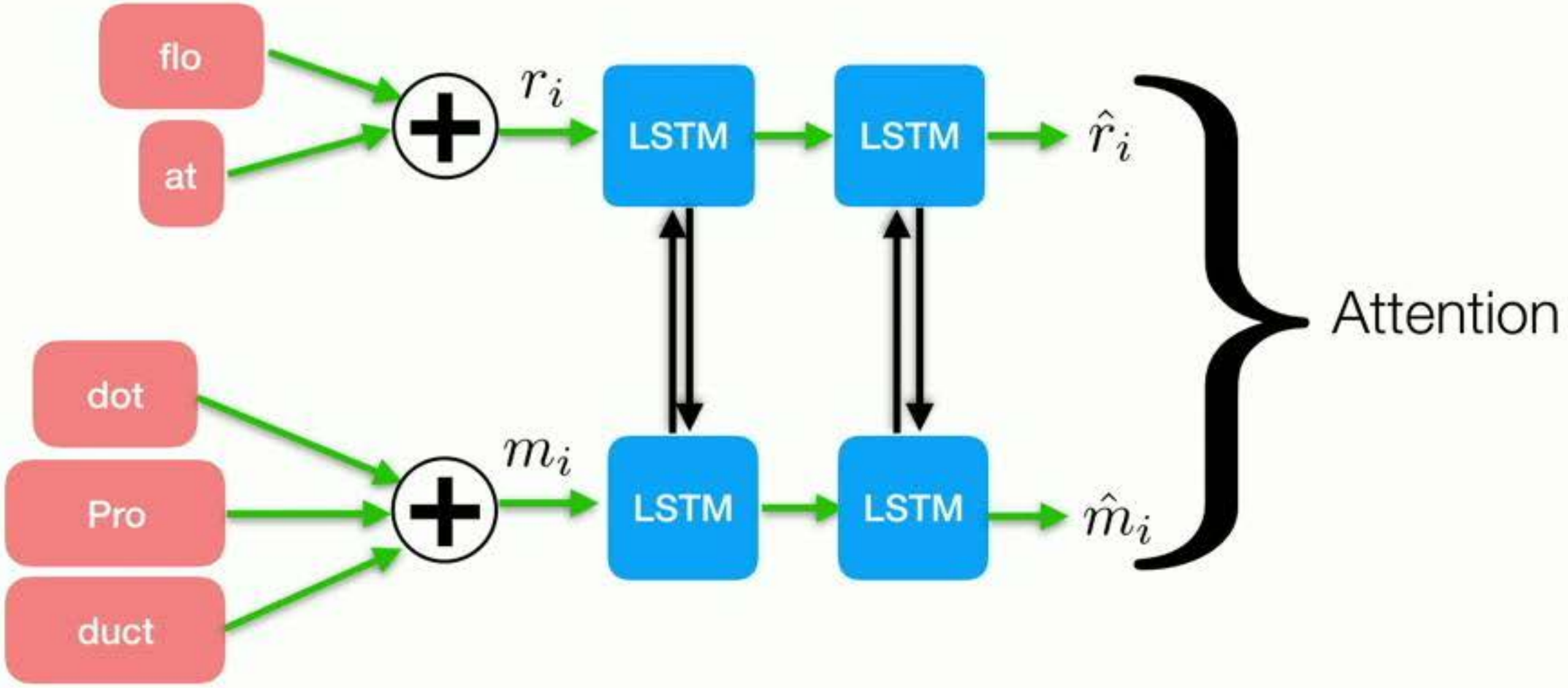


Encoding the Environment

flo at

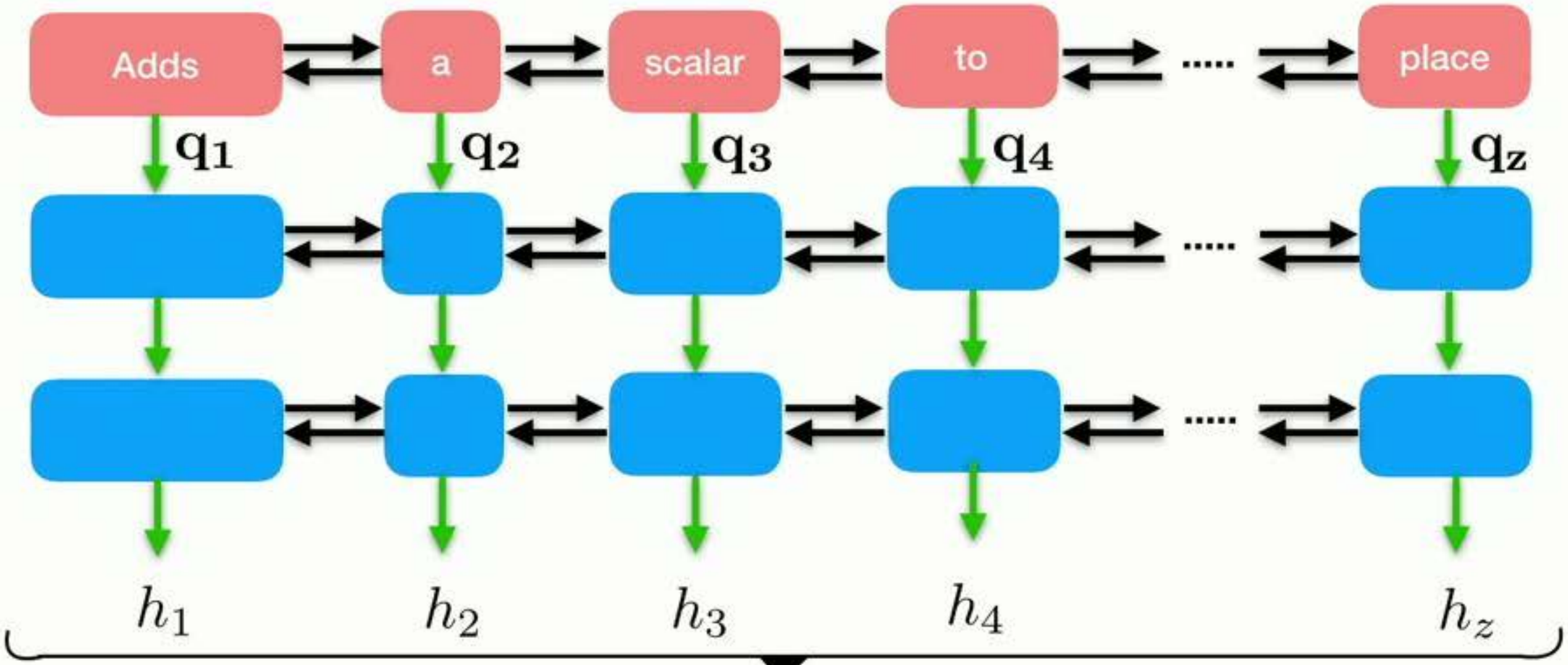
dot Pro duct

$$\mathbf{r_i} = Avg(\mathbf{r_{i1}}, \ldots, \mathbf{r_{ij}})$$
$$\mathbf{m_i} = Avg(\mathbf{m_{i1}}, \ldots, \mathbf{m_{ij}})$$
$$\hat{r}_i, \hat{m}_i = f(\mathbf{r_i}, \mathbf{m_i})$$



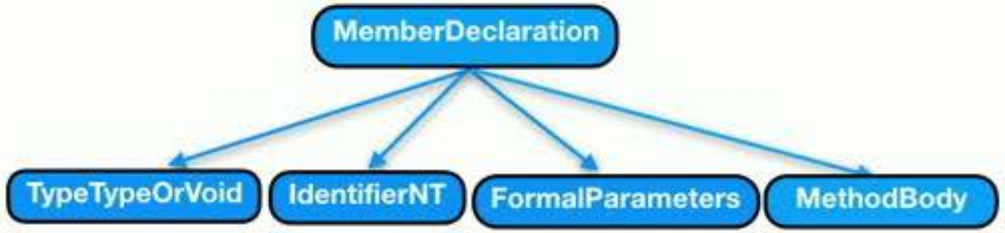
Encoding the NL Query

Adds a scalar to this vector in place.



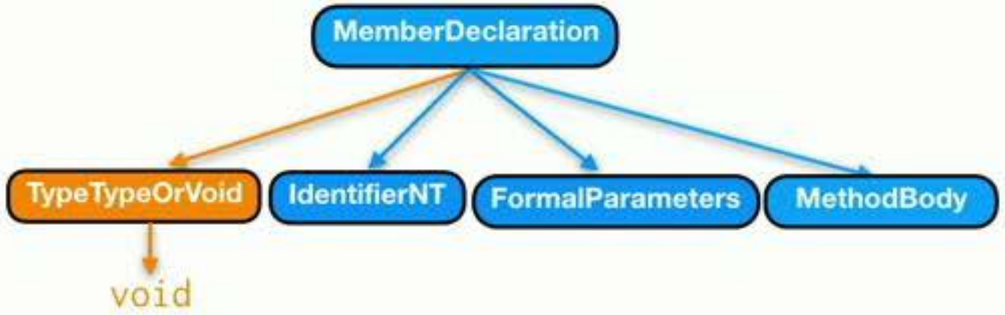
$$h_1, \cdots, h_z = \text{Bi-LSTM}(\mathbf{q_1}, \ldots, \mathbf{q_z})$$

Decoder RNN


MemberDeclaration
TypeTypeOrVoid
IdentifierNT
FormalParameters
MethodBody

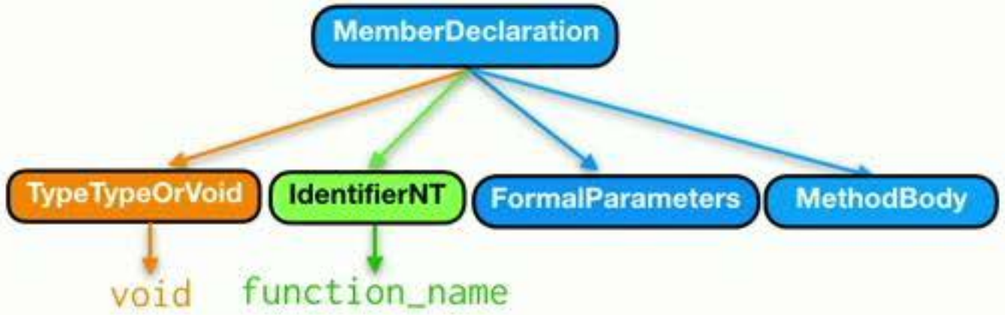

LSTM
MemberDeclaration → TypeTypeOrVoid IdentifierNT
FormalParameters MethodBody

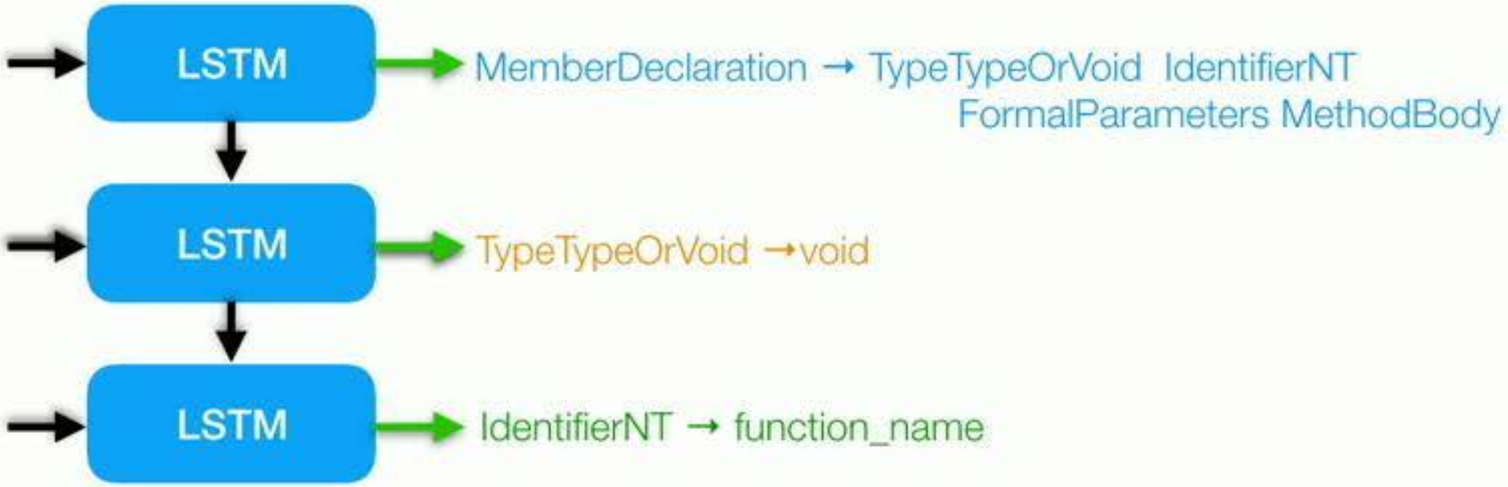
Decoder RNN


MemberDeclaration
TypeTypeOrVoid
IdentifierNT
FormalParameters
MethodBody
void

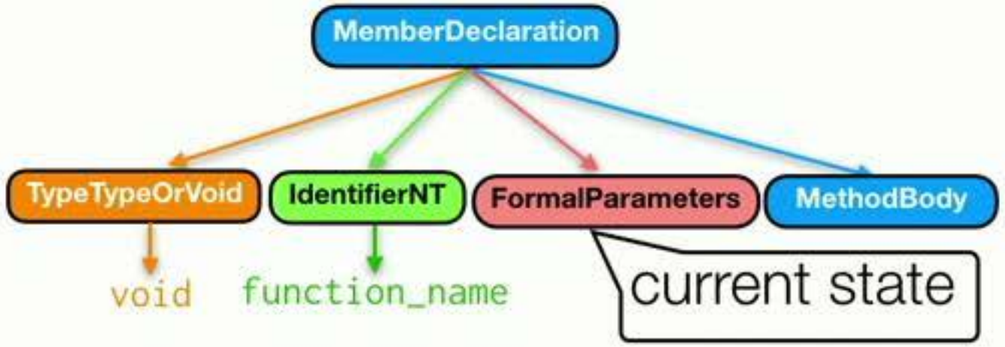

LSTM
MemberDeclaration → TypeTypeOrVoid IdentifierNT
FormalParameters MethodBody
LSTM
TypeTypeOrVoid →void

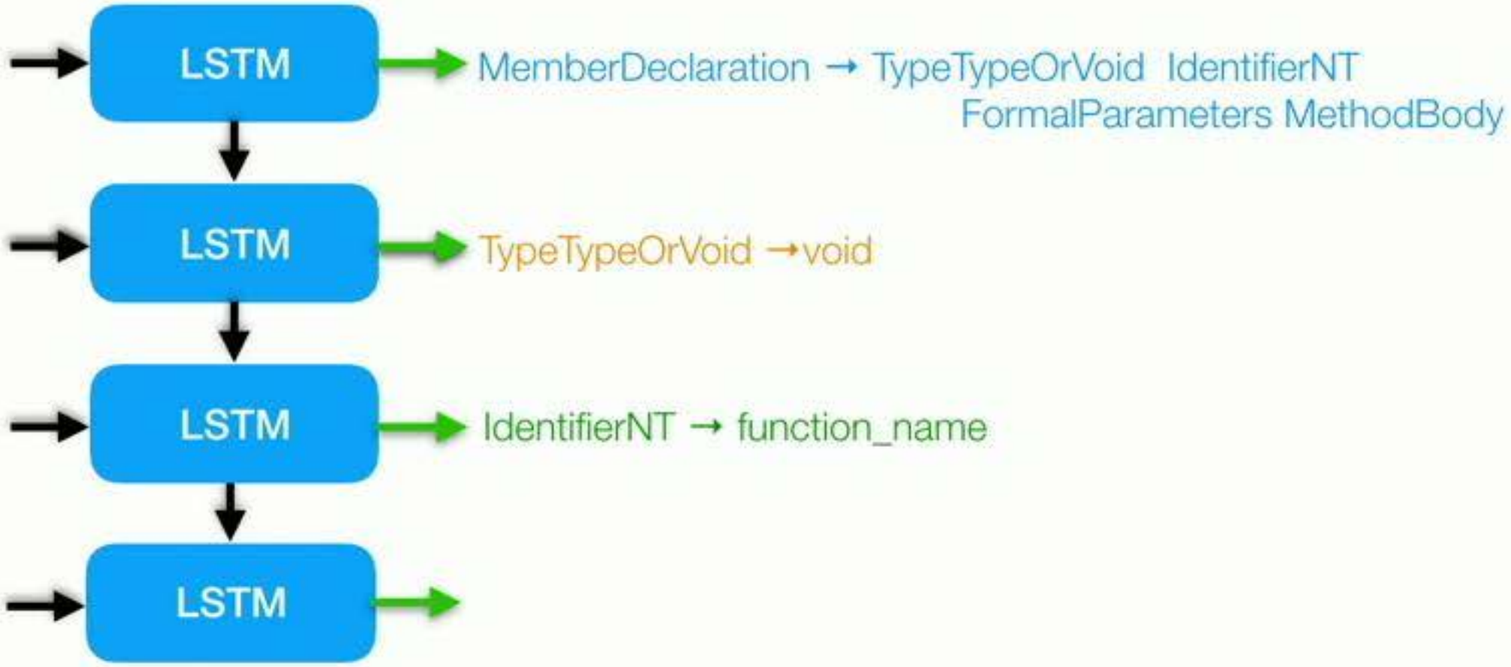
Decoder RNN


MemberDeclaration
TypeTypeOrVoid
IdentifierNT
FormalParameters
MethodBody
void
function_name

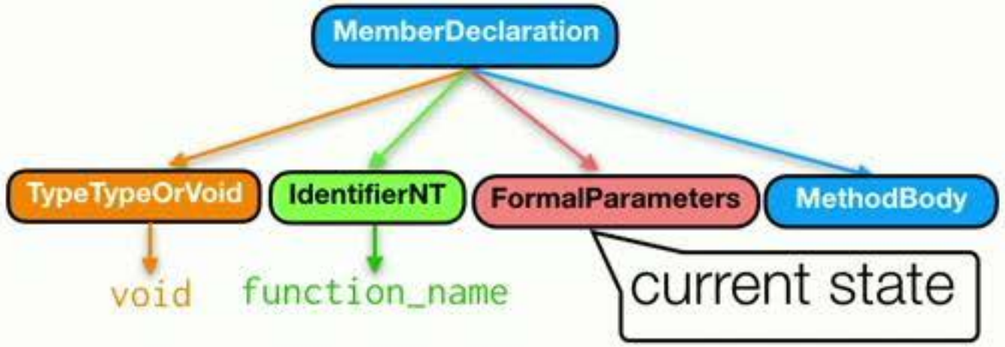

LSTM
MemberDeclaration → TypeTypeOrVoid IdentifierNT
FormalParameters MethodBody
LSTM
TypeTypeOrVoid →void
LSTM
IdentifierNT → function_name

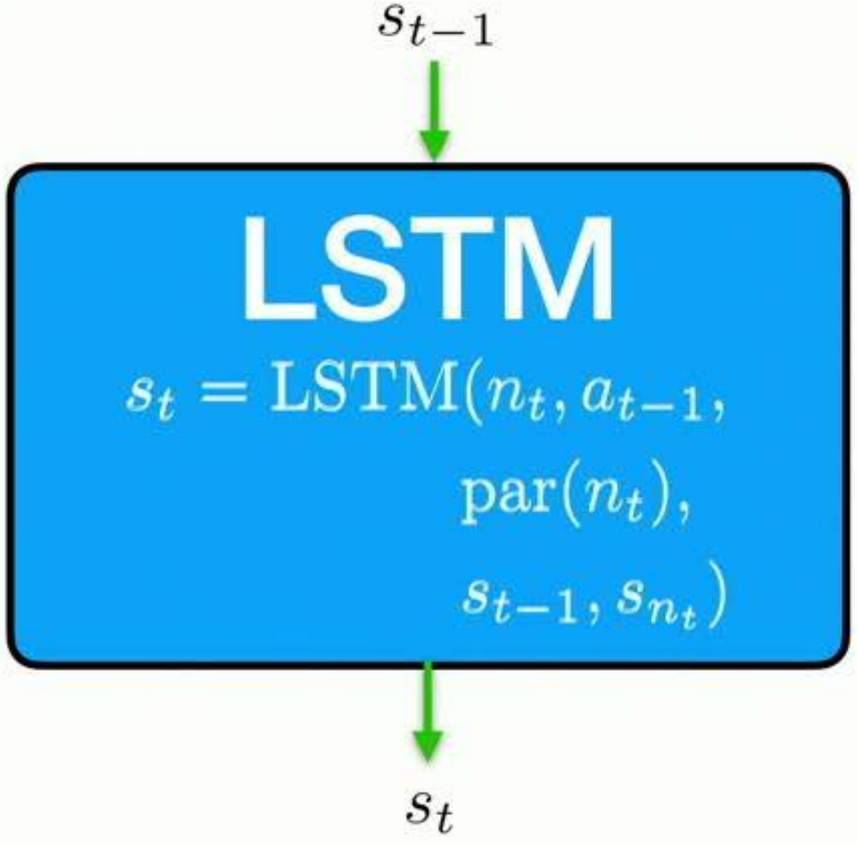
Decoder RNN


MemberDeclaration
TypeTypeOrVoid
IdentifierNT
FormalParameters
MethodBody
void
function_name
current state

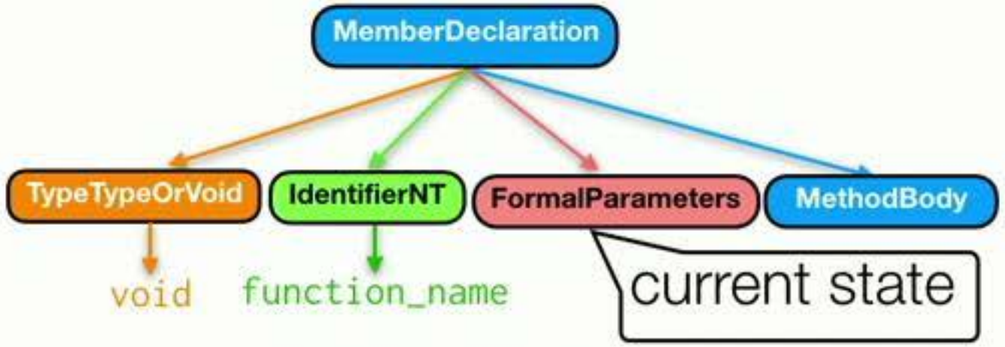

LSTM
MemberDeclaration → TypeTypeOrVoid IdentifierNT FormalParameters MethodBody
LSTM
TypeTypeOrVoid →void
LSTM
IdentifierNT → function_name
LSTM

Decoder RNN

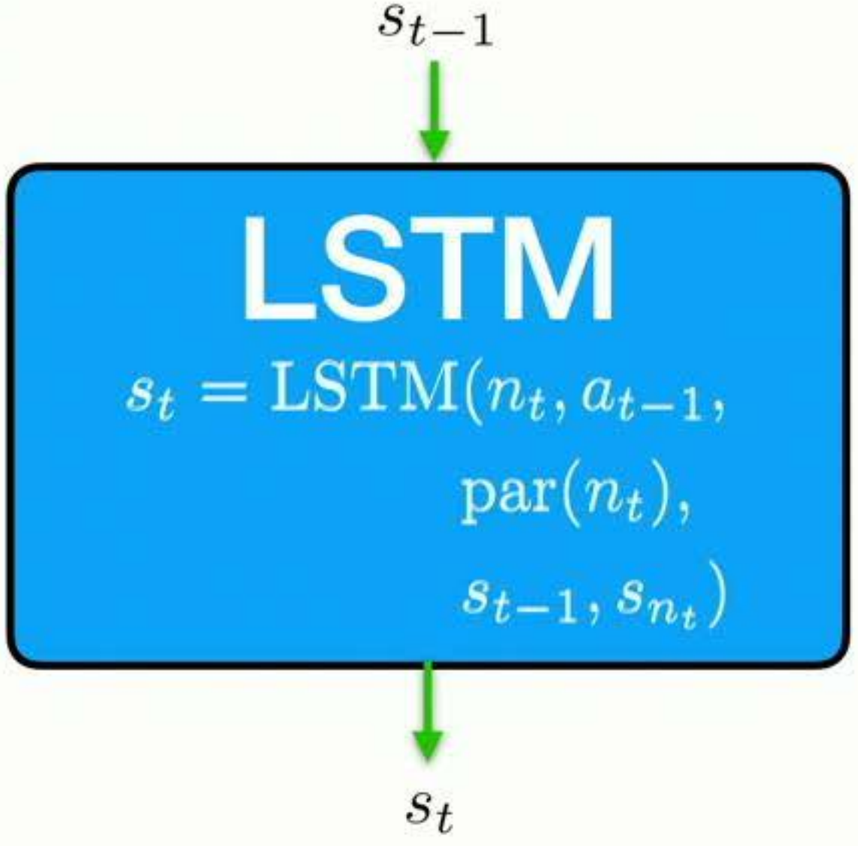

MemberDeclaration
TypeTypeOrVoid
IdentifierNT
FormalParameters
MethodBody
void
function_name
current state


s_{t-1}
LSTM
$s_t = \text{LSTM}(n_t, a_{t-1}, \text{par}(n_t), s_{t-1}, s_{n_t})$
s_t

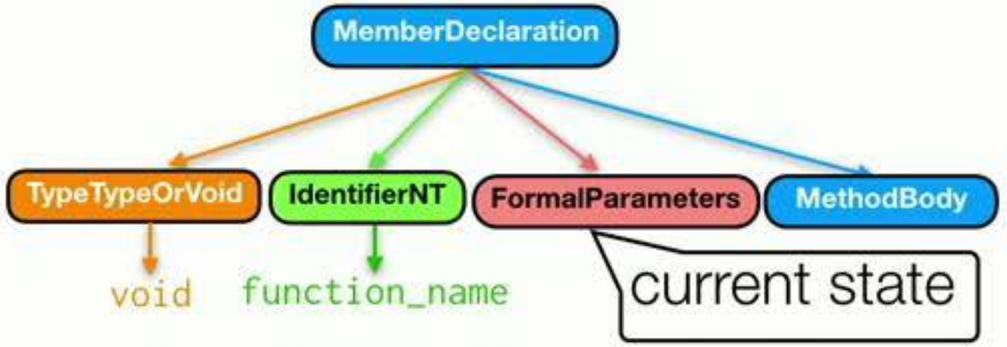
Decoder RNN



FormalParameters (n_t)



Decoder RNN



FormalParameters (n_t)

IdentifierNT → function_name (a_{t-1})

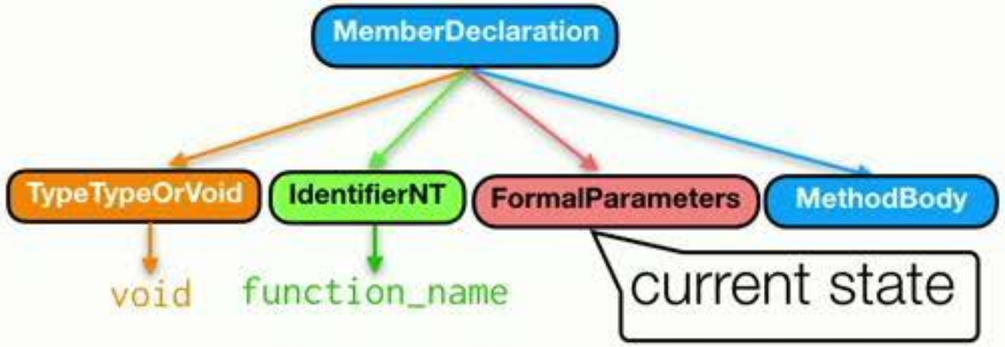
s_{t-1}

LSTM

$$s_t = \text{LSTM}(n_t, a_{t-1}, \text{par}(n_t), s_{t-1}, s_{n_t})$$

s_t

Decoder RNN



FormalParameters (n_t)

IdentifierNT → function_name (a_{t-1})

MemberDeclaration → TypeTypeOrVoid identifierNT FormalParameters MethodBody $(par(n_t))$

Feed parent action

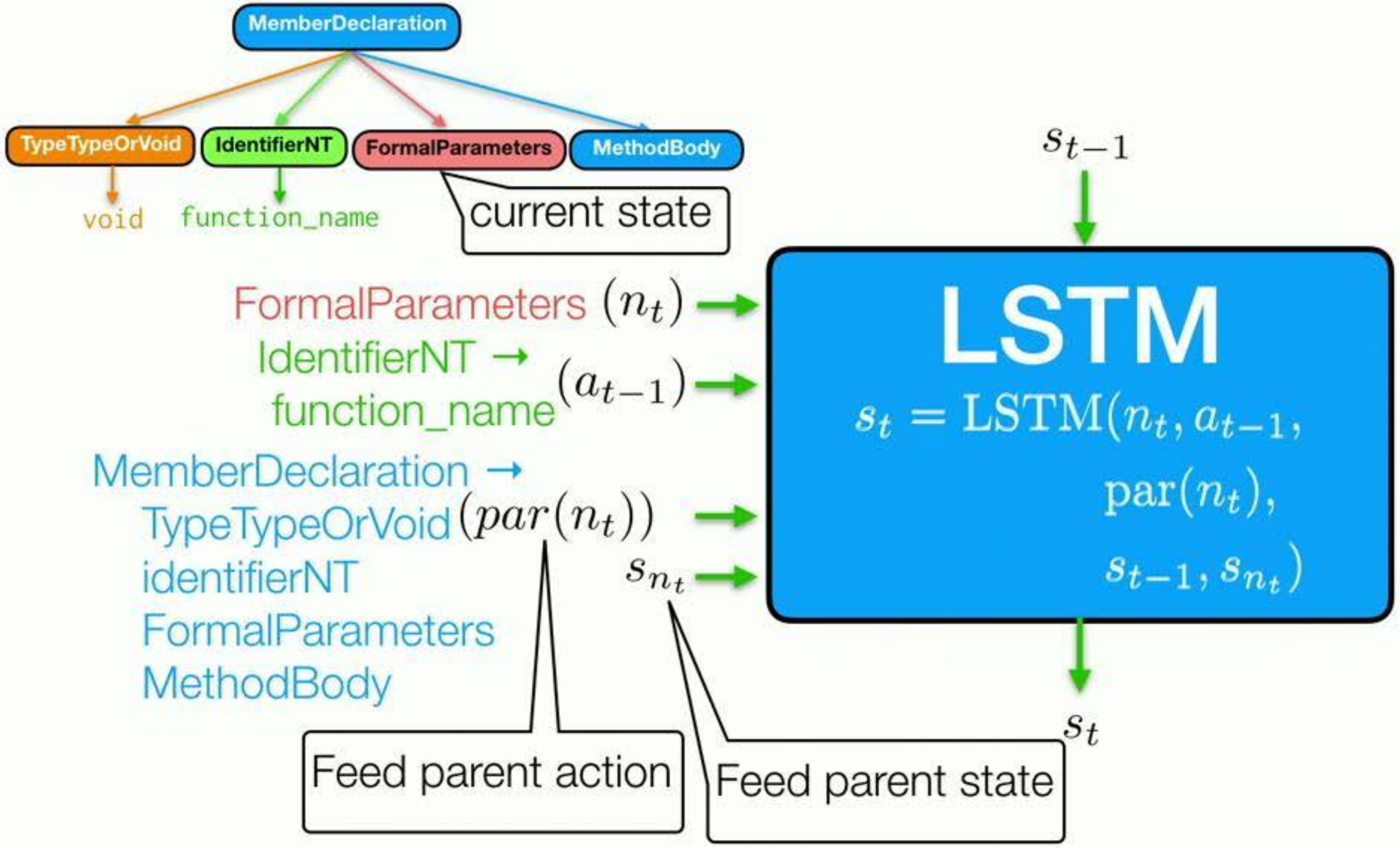
s_{t-1}

LSTM

$$s_t = \text{LSTM}(n_t, a_{t-1}, \text{par}(n_t), s_{t-1}, s_{n_t})$$

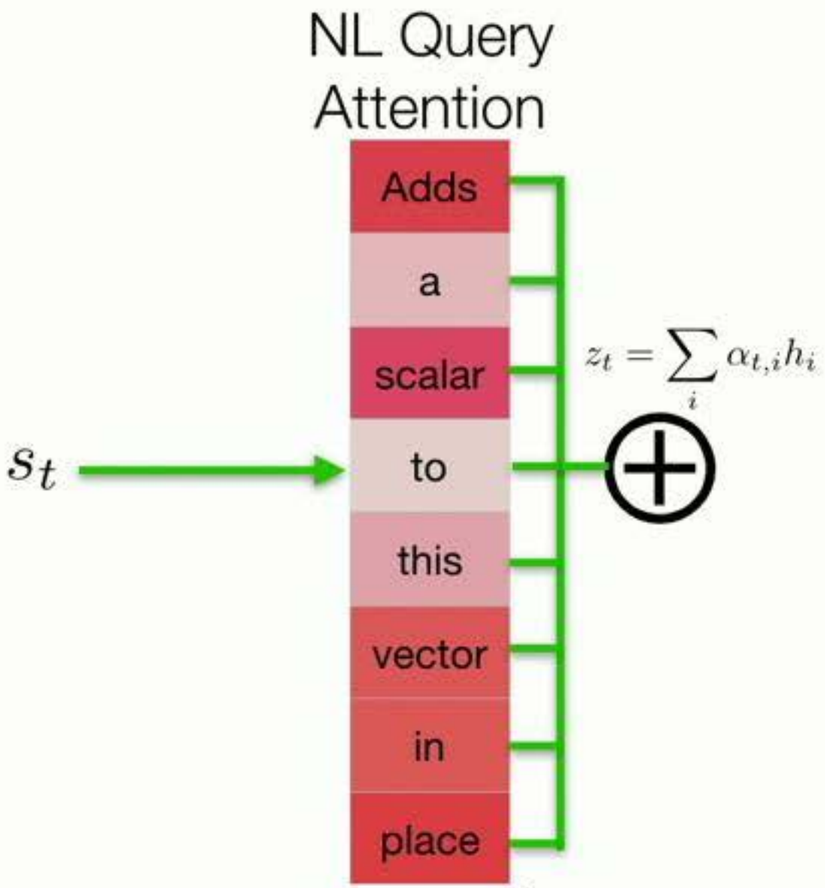
s_t

Decoder RNN

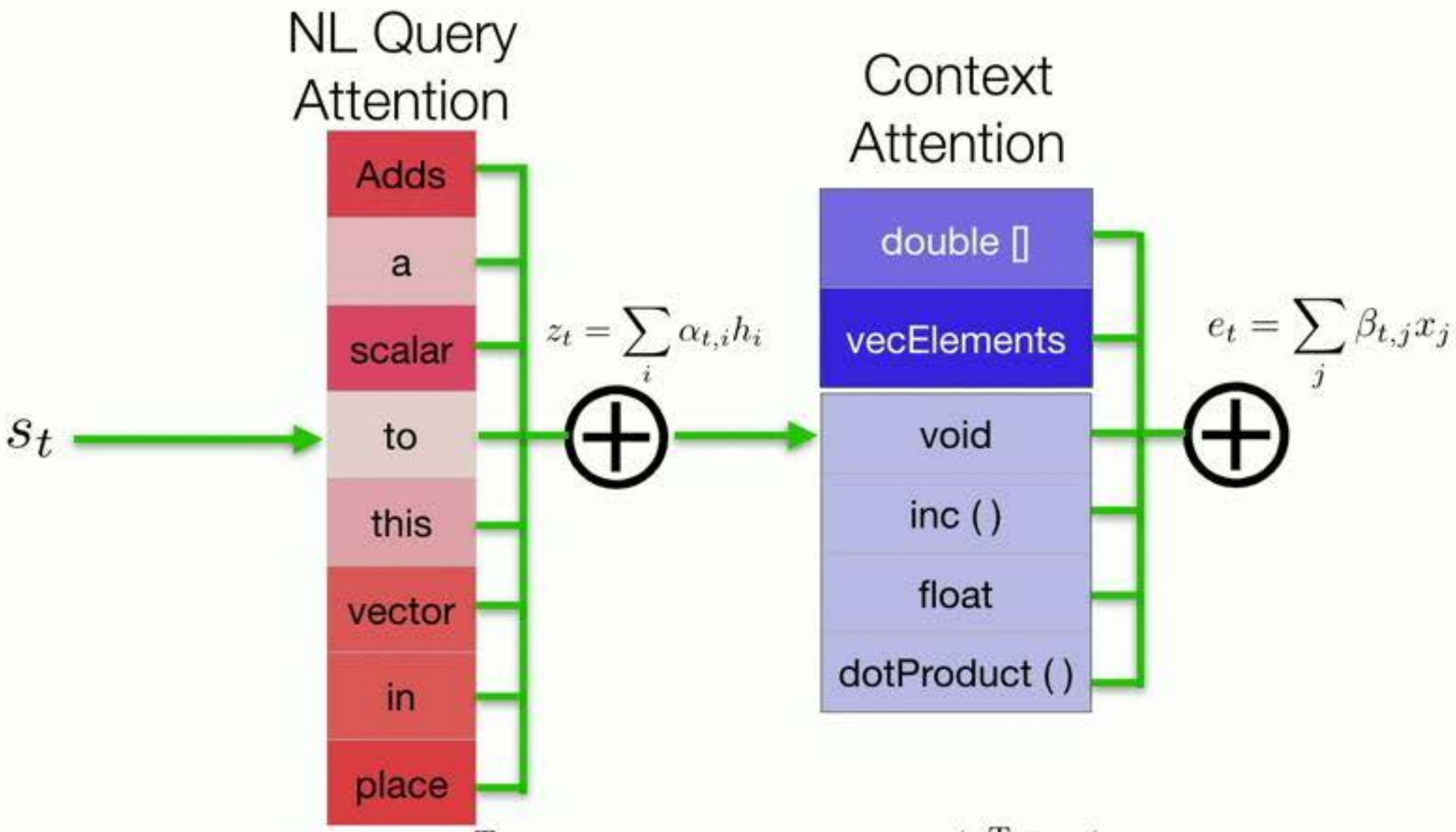

MemberDeclaration
TypeTypeOrVoid
IdentifierNT
FormalParameters
MethodBody
void
function_name
current state
s_{t-1}
FormalParameters (n_t)
IdentifierNT → function_name (a_{t-1})
MemberDeclaration → TypeTypeOrVoid identifierNT FormalParameters MethodBody $(par(n_t))$
s_{n_t}
LSTM
$s_t = \text{LSTM}(n_t, a_{t-1}, \text{par}(n_t), s_{t-1}, s_{n_t})$
s_t
Feed parent action
Feed parent state

Decoder Attention Mechanisms

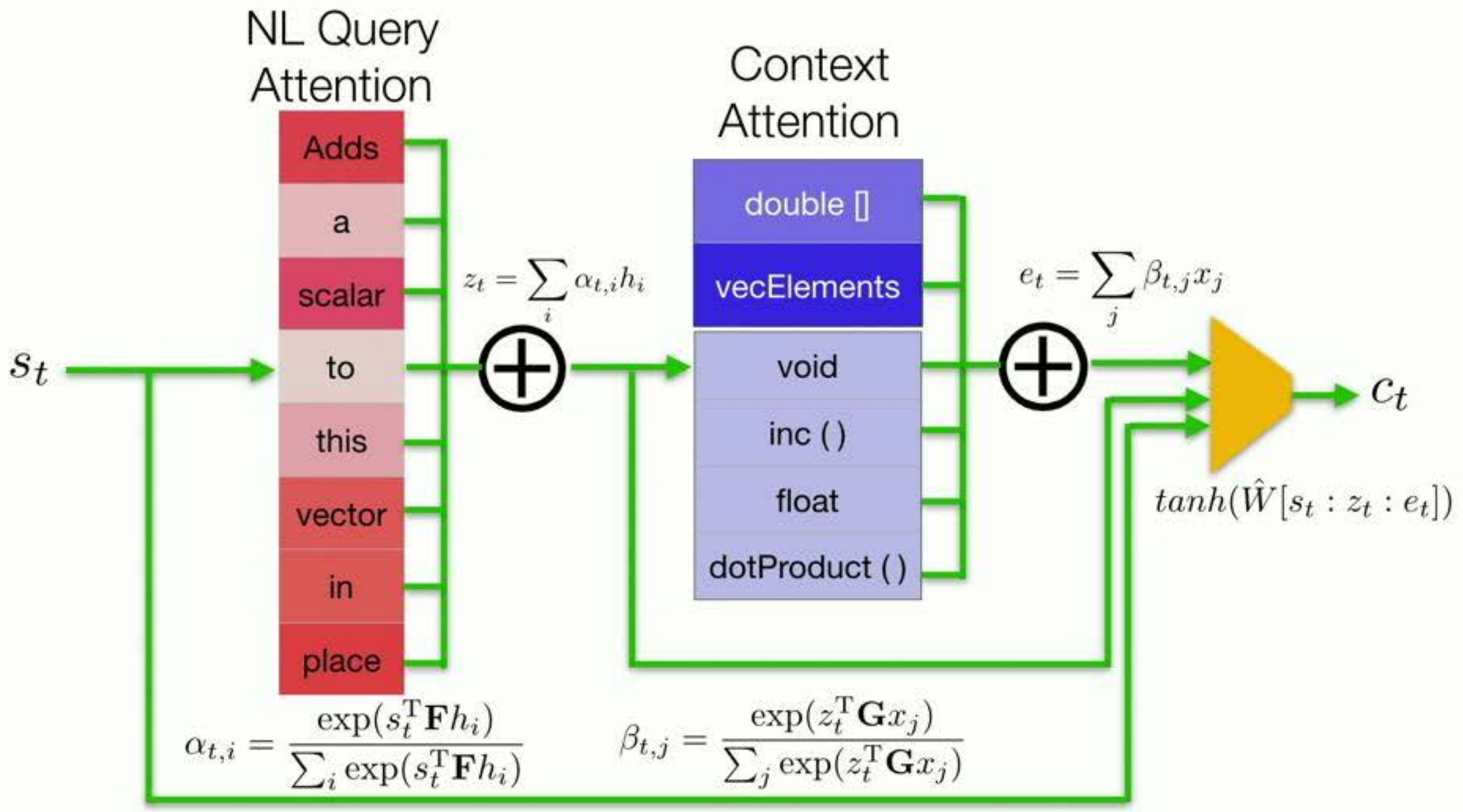
Decoder Attention Mechanisms



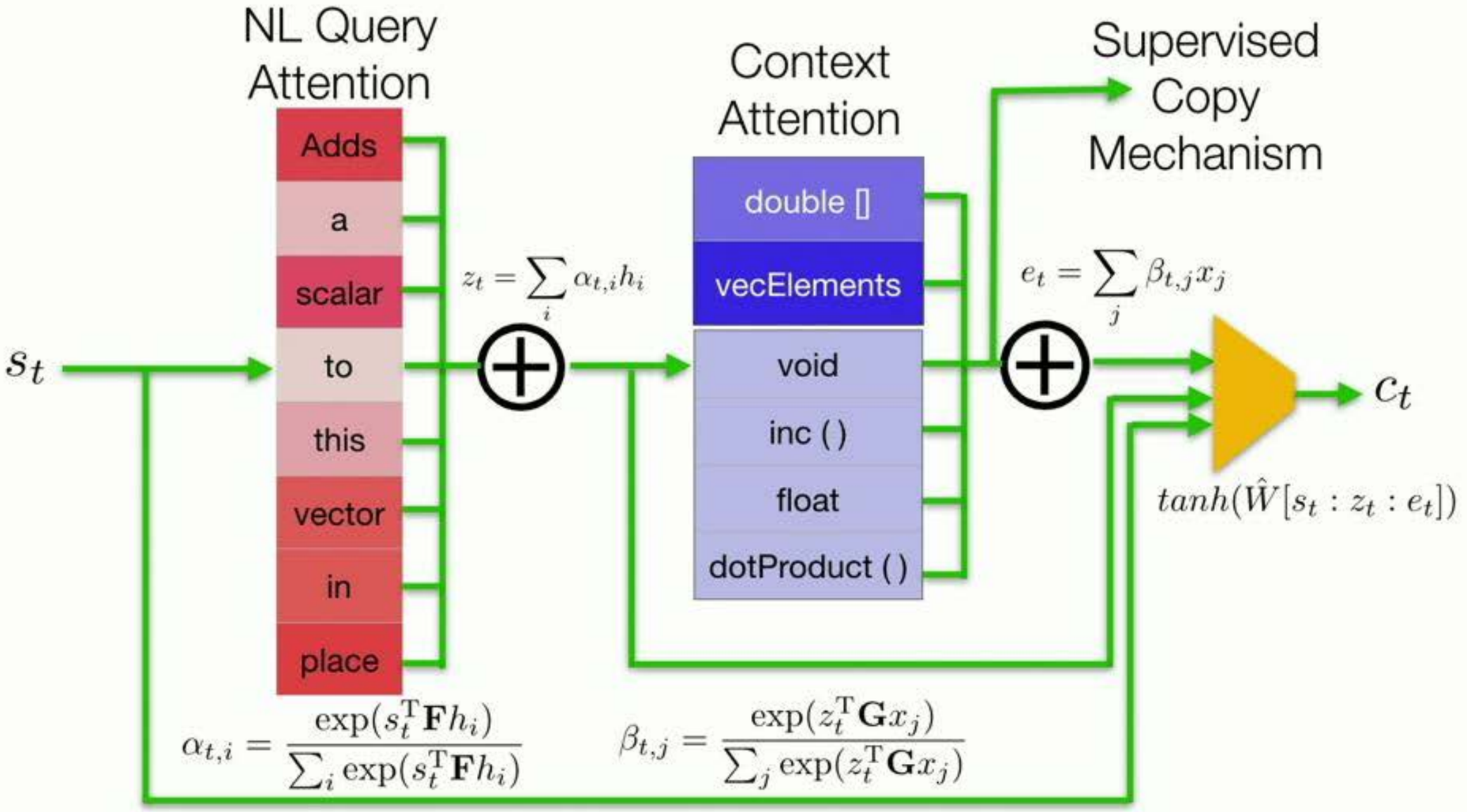
Decoder Attention Mechanisms



Decoder Attention Mechanisms

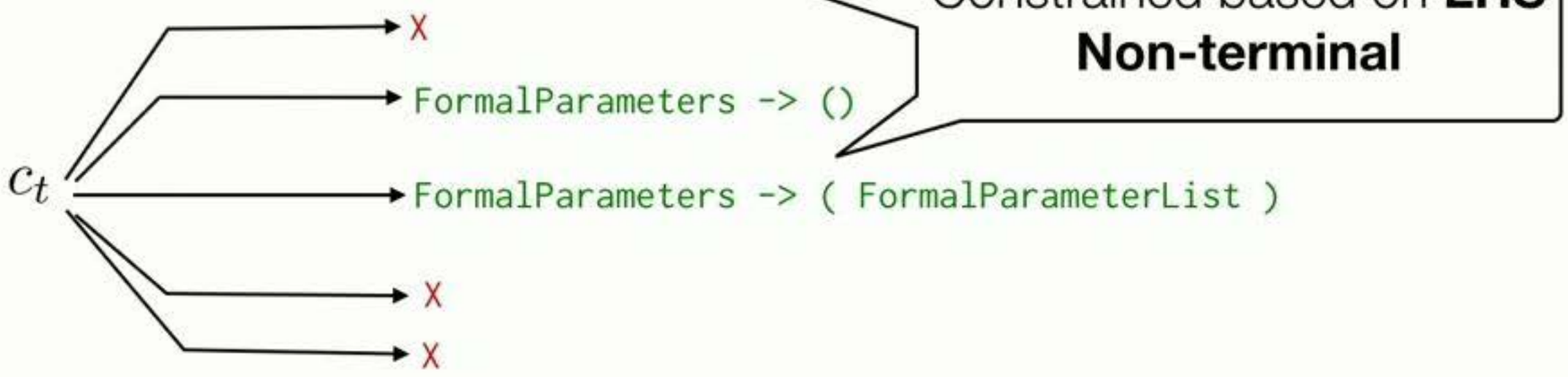


Decoder Attention Mechanisms

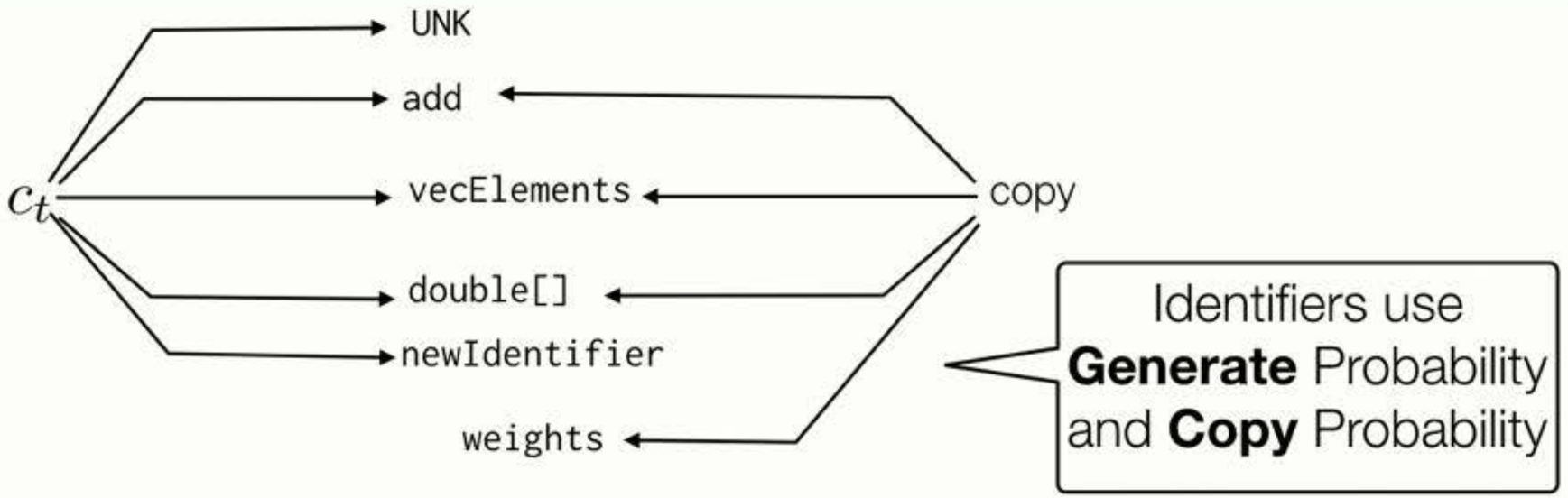


Output Generation

$$p(a_t|a_{<t}) \propto exp(W^{n_t} c_t)$$



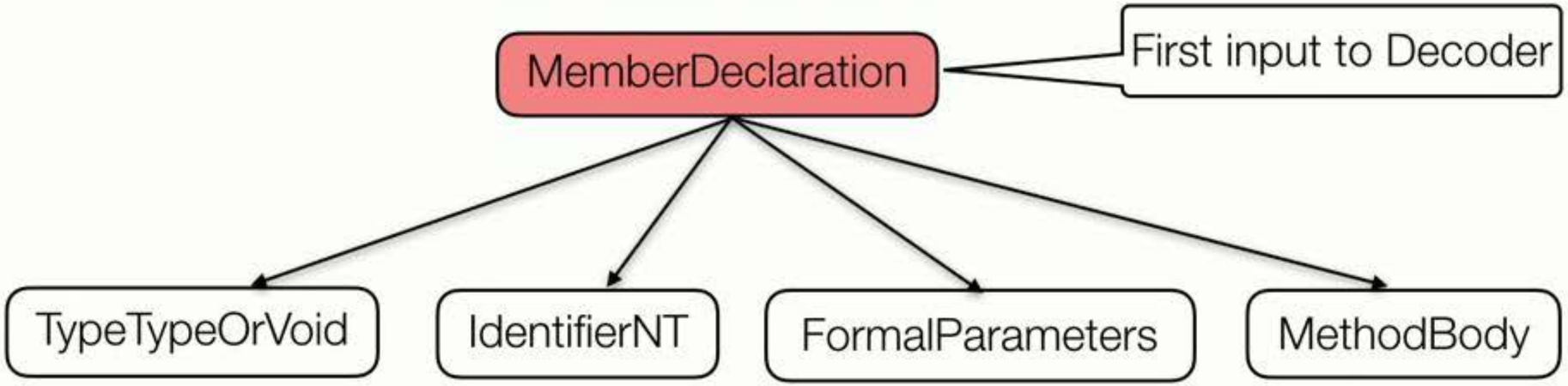
Output Generation



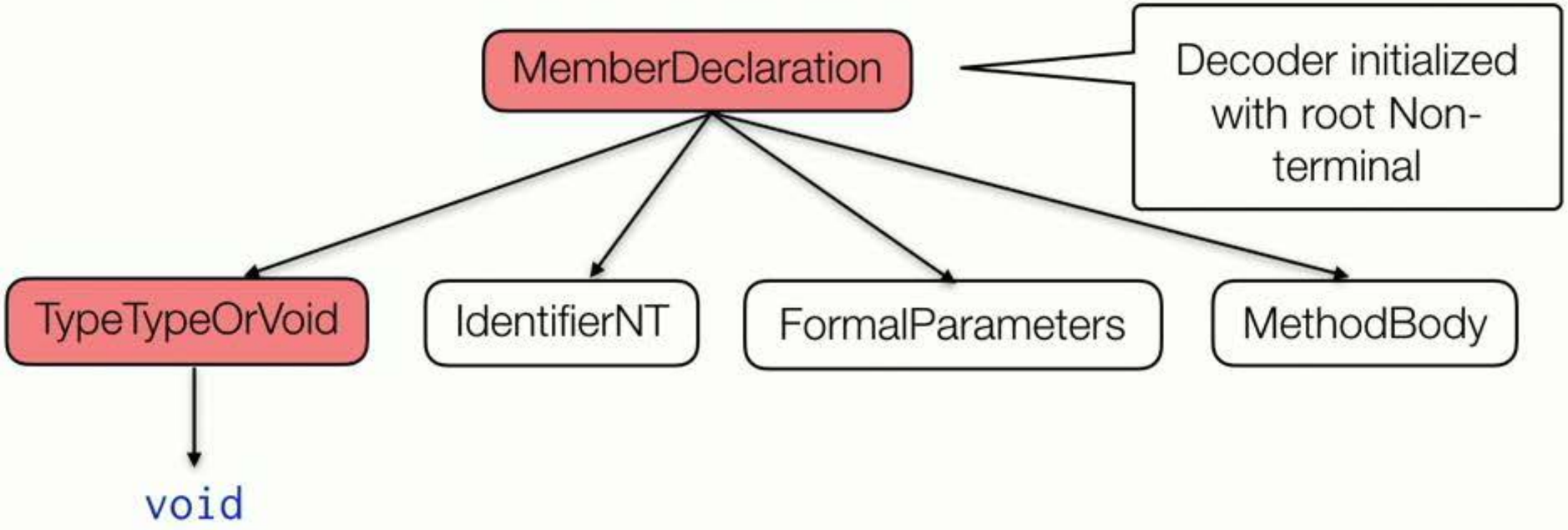
$$p_{copy} = \sigma(b^T c_t)$$

$$p_{identifier} = p_{copy} \times \beta_{identifier} + (1 - p_{copy}) \times p_{IdentifierNT \to identifier}$$

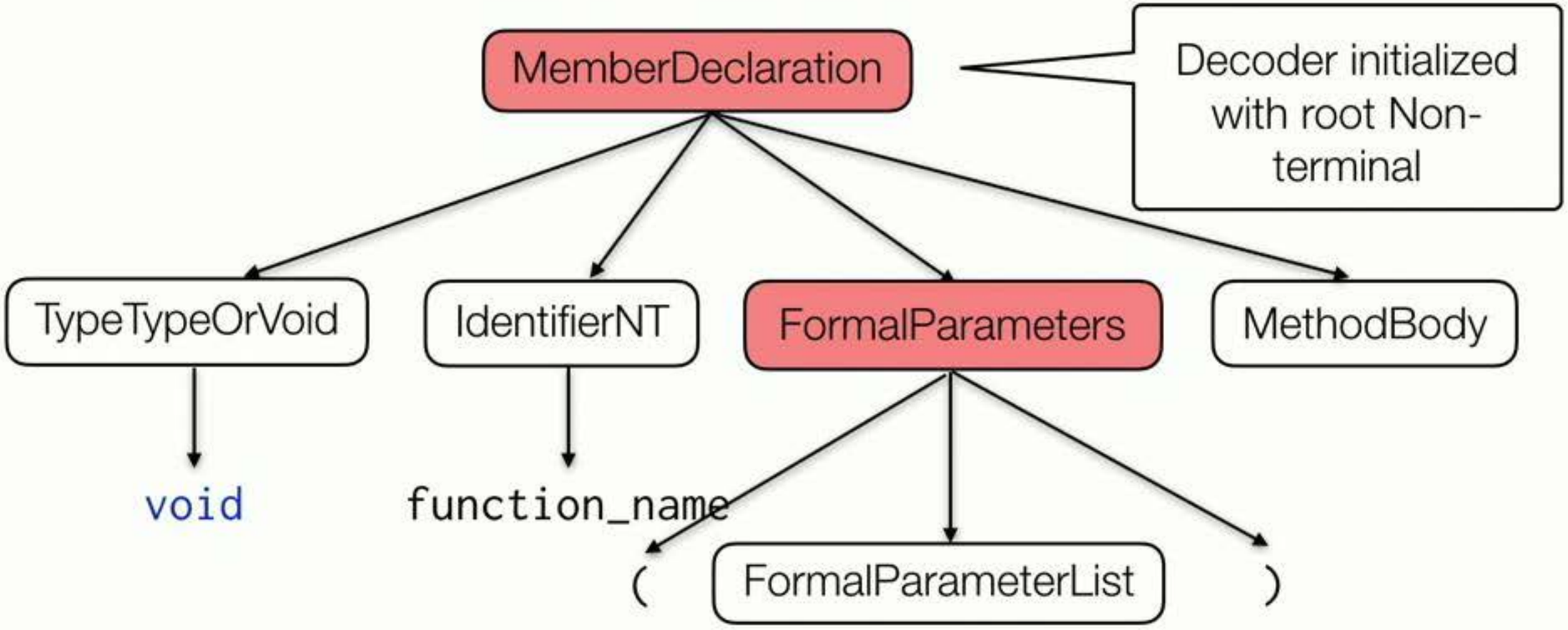
Stack-based Decoding



Stack-based Decoding

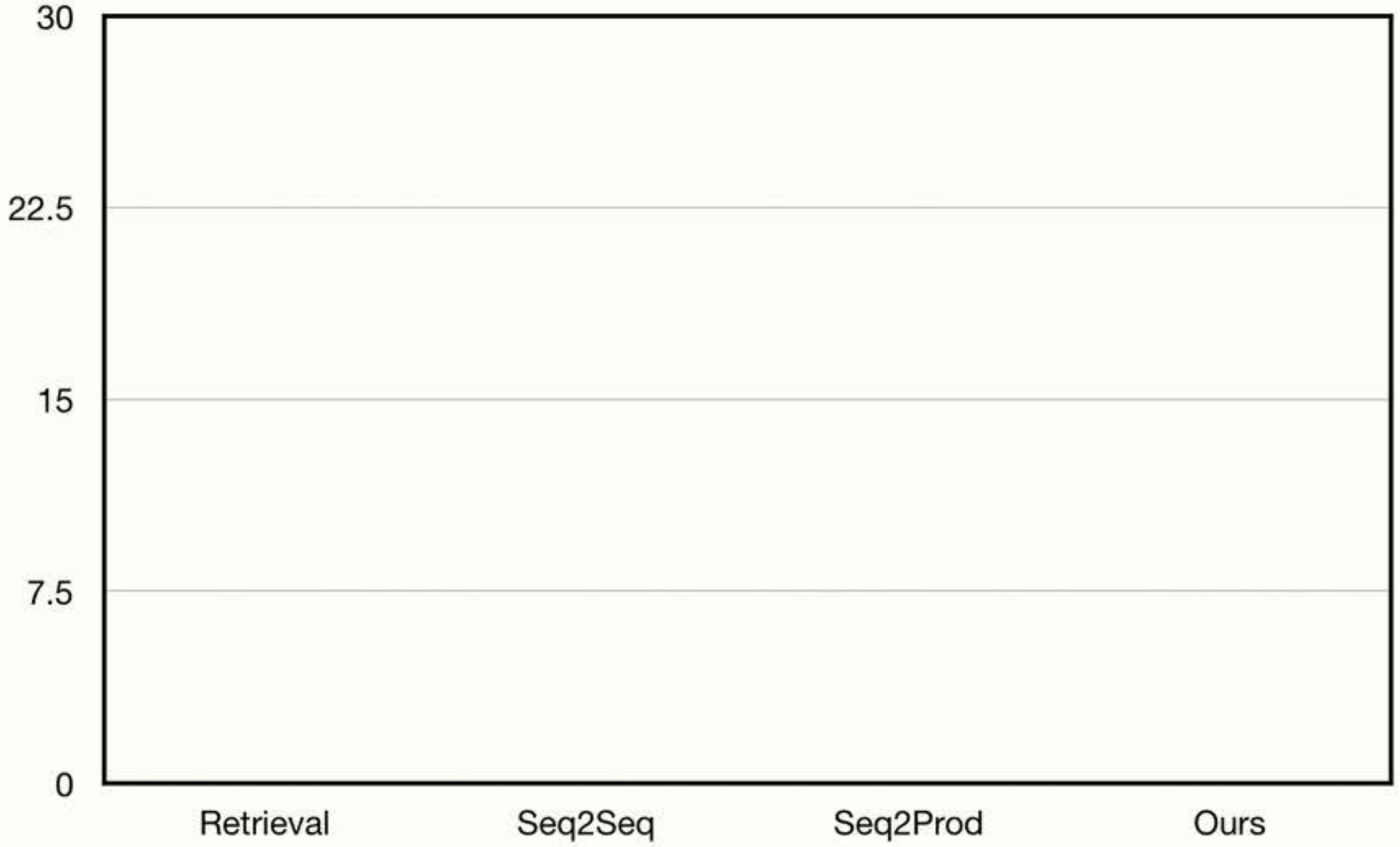


Stack-based Decoding

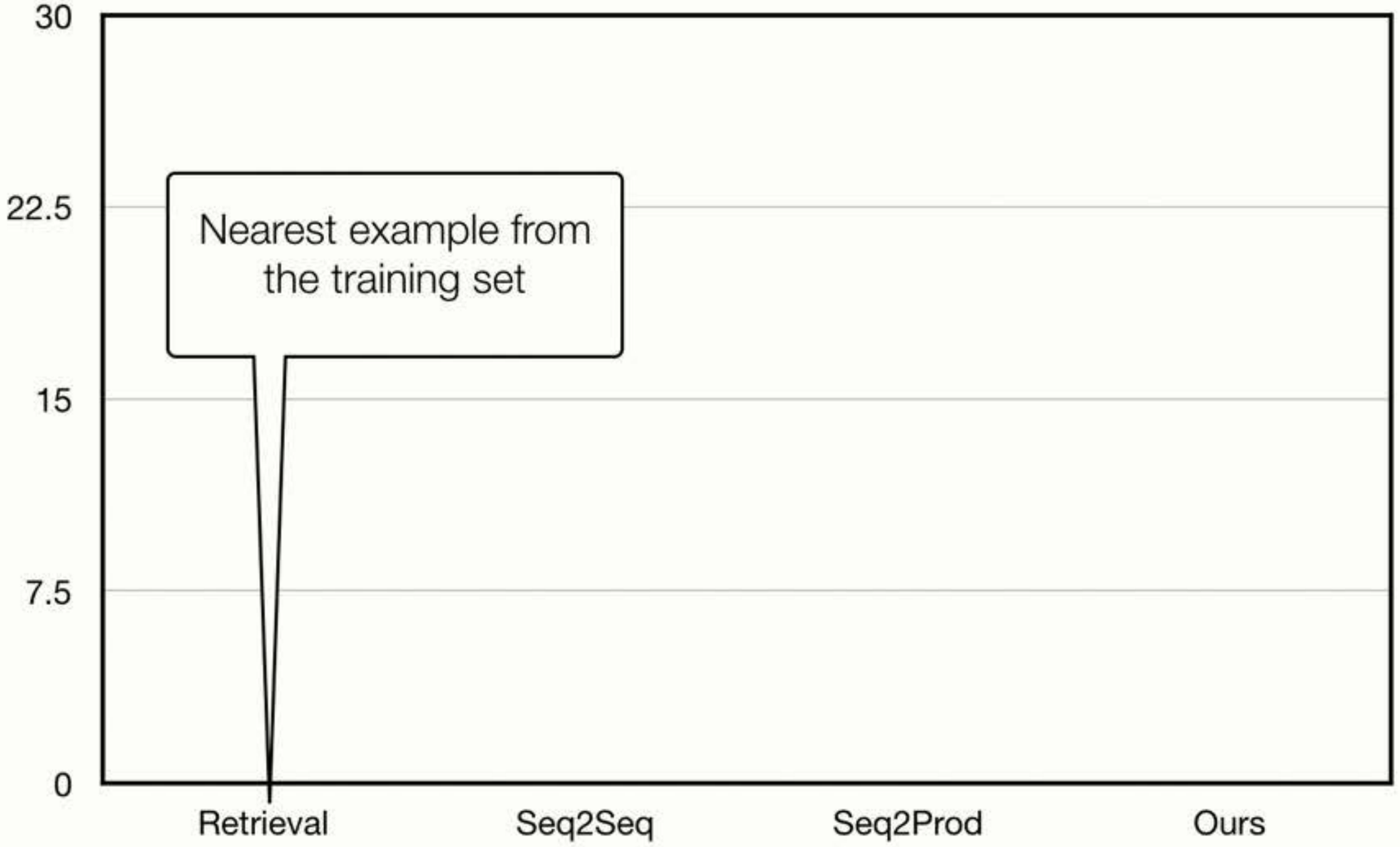


Use Beam Search over Production Rules

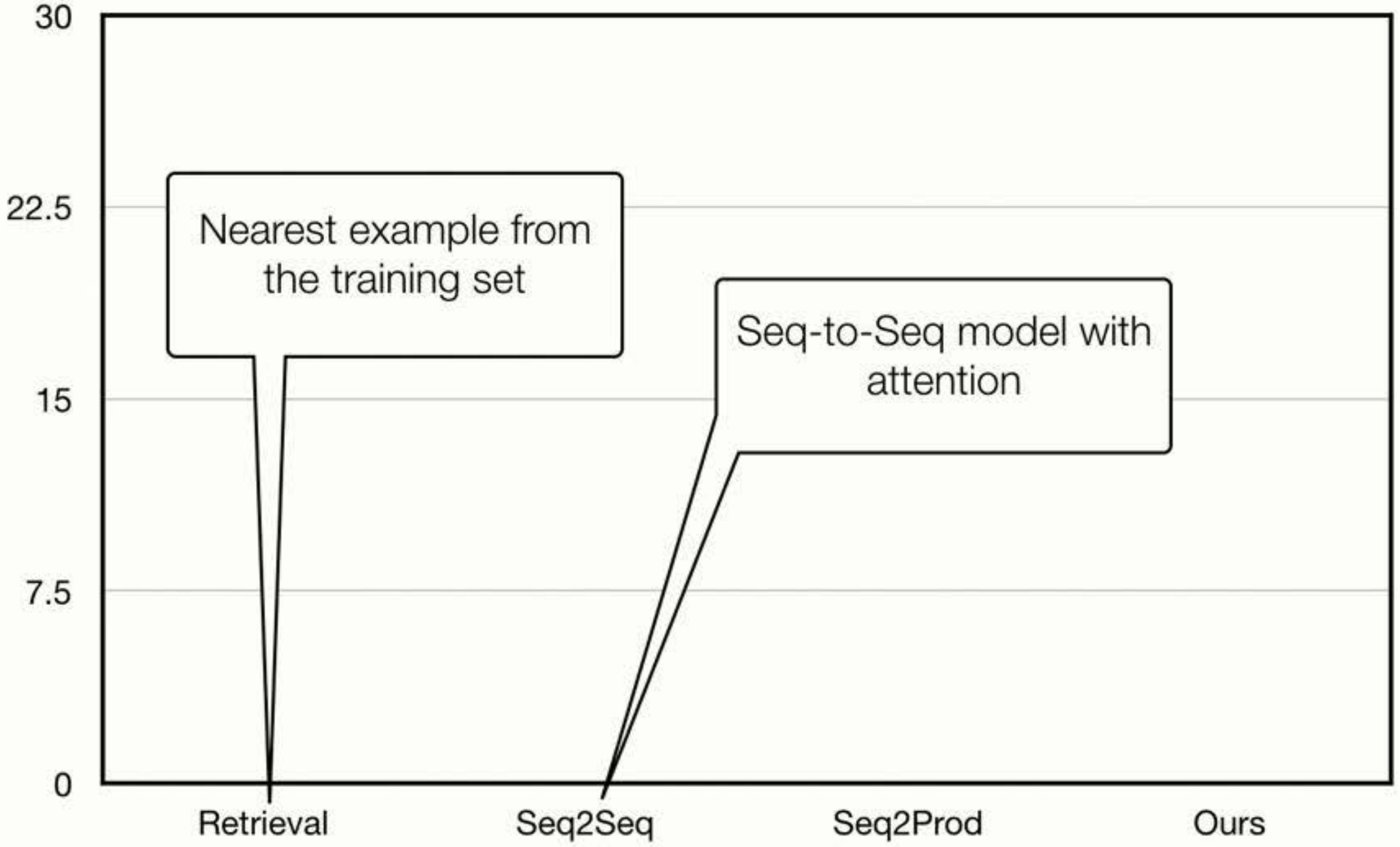
Baseline Models



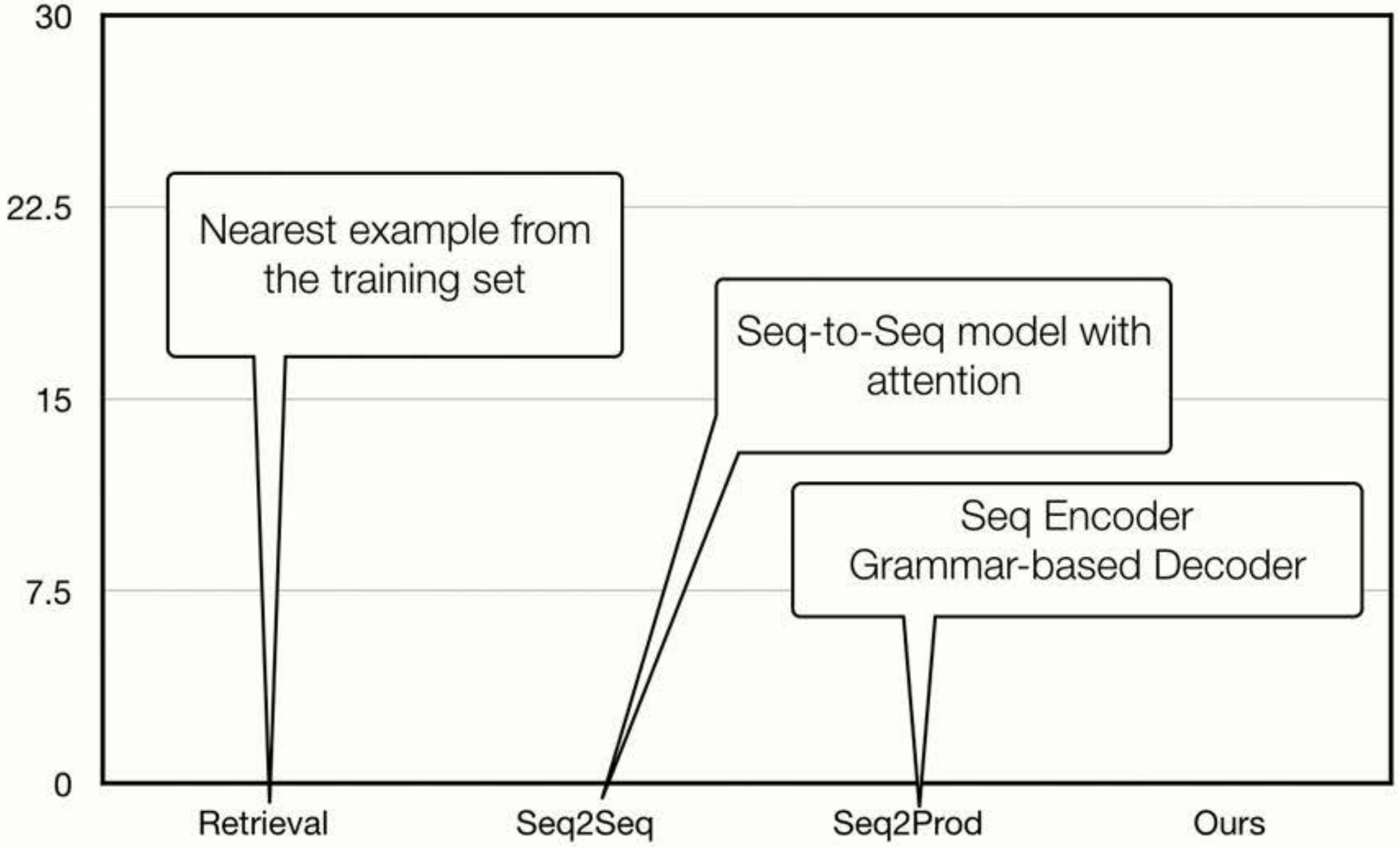
Baseline Models



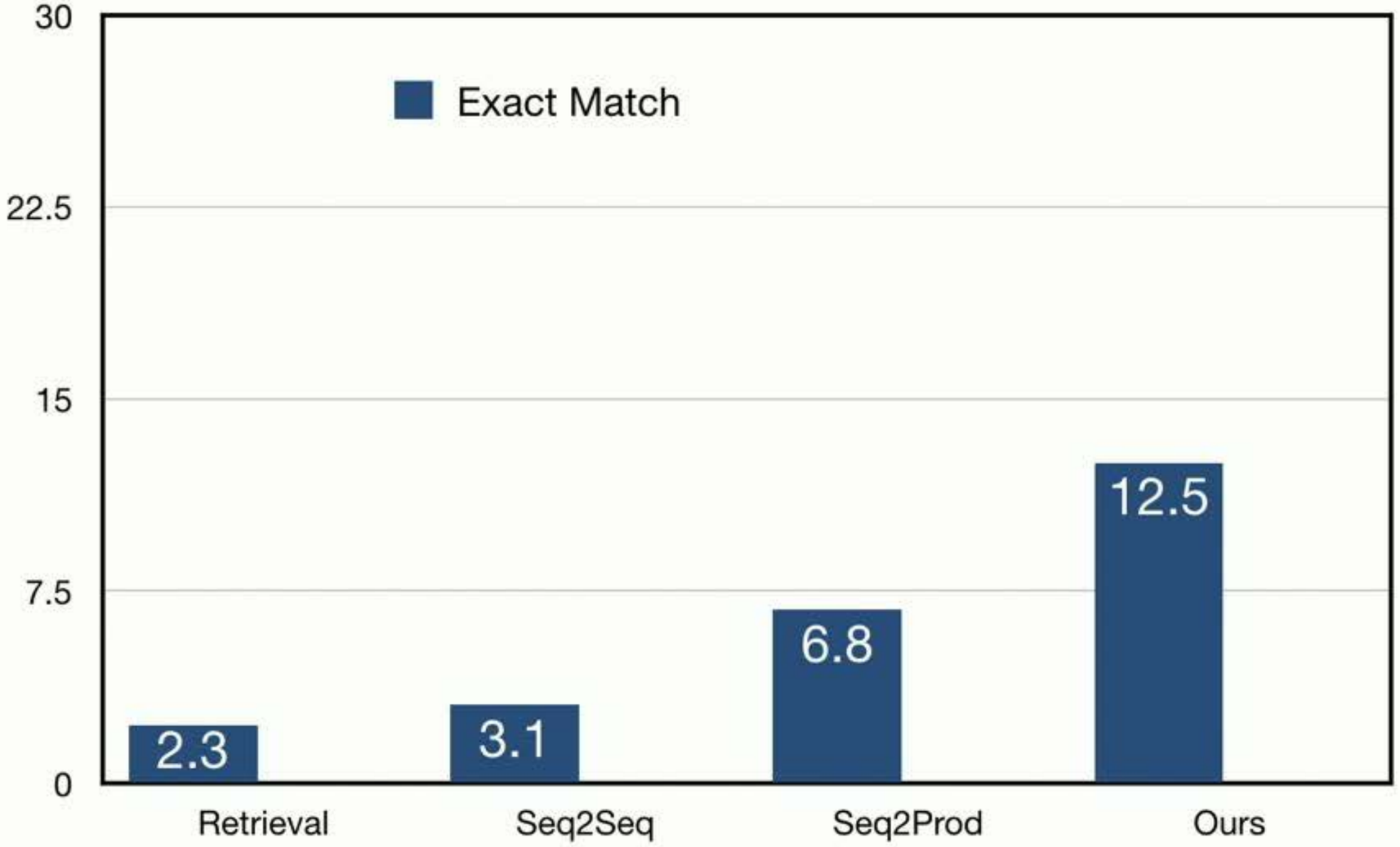
Baseline Models



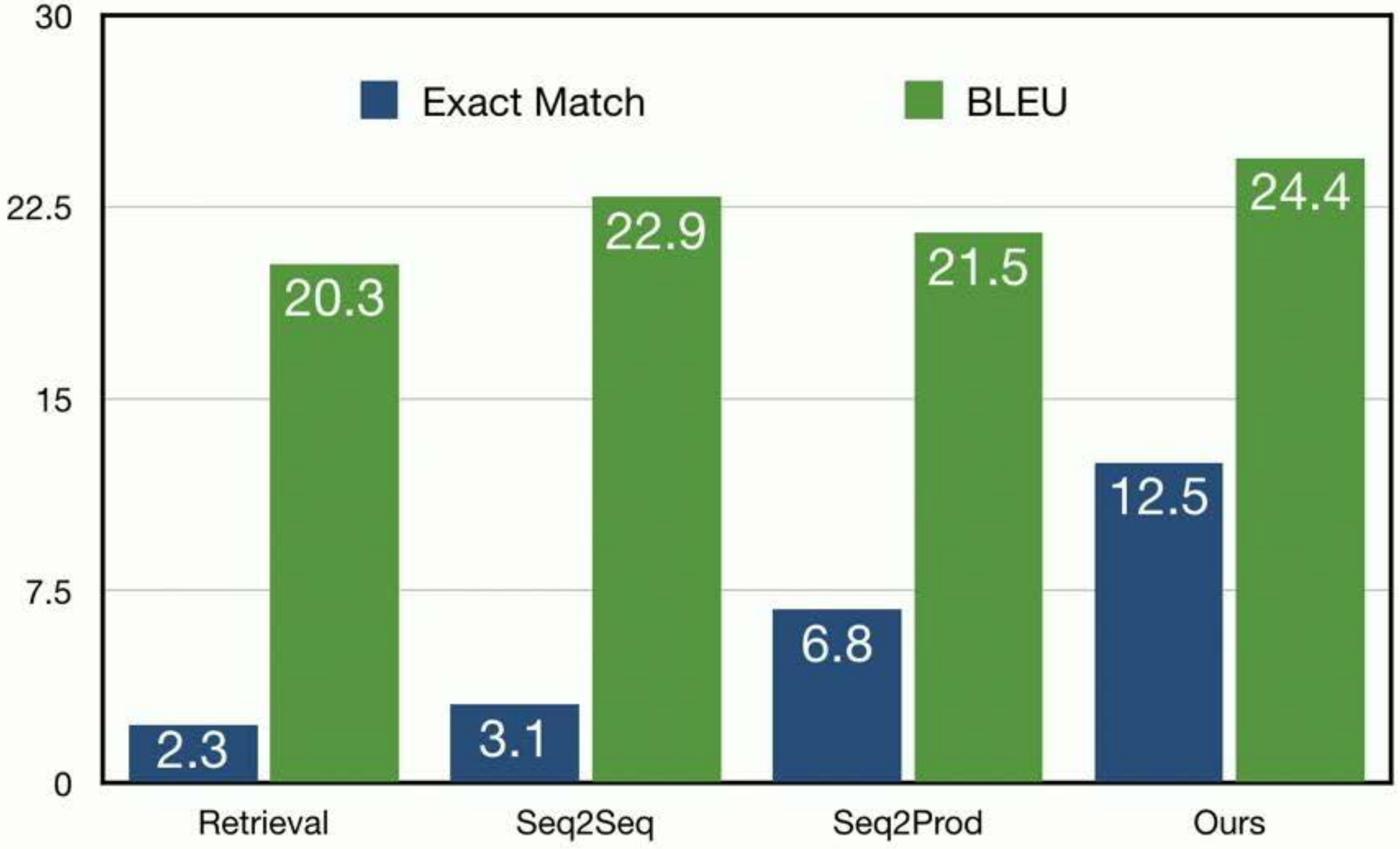
Baseline Models



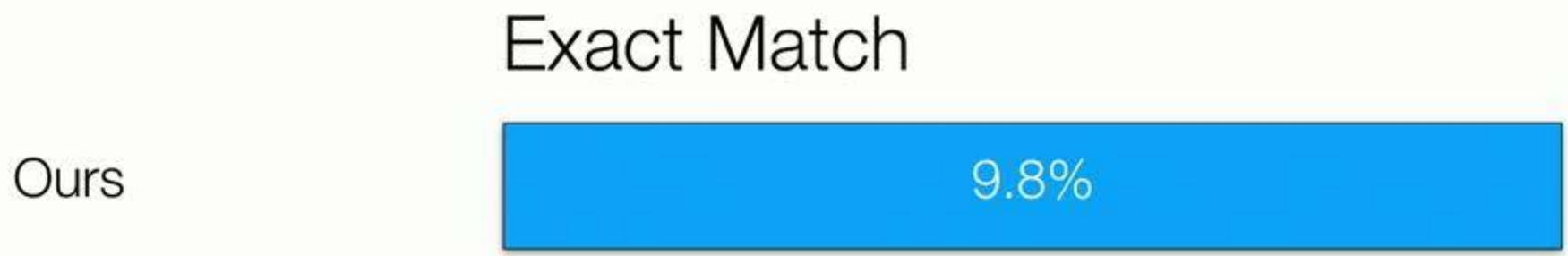
Exact Match and BLEU Score



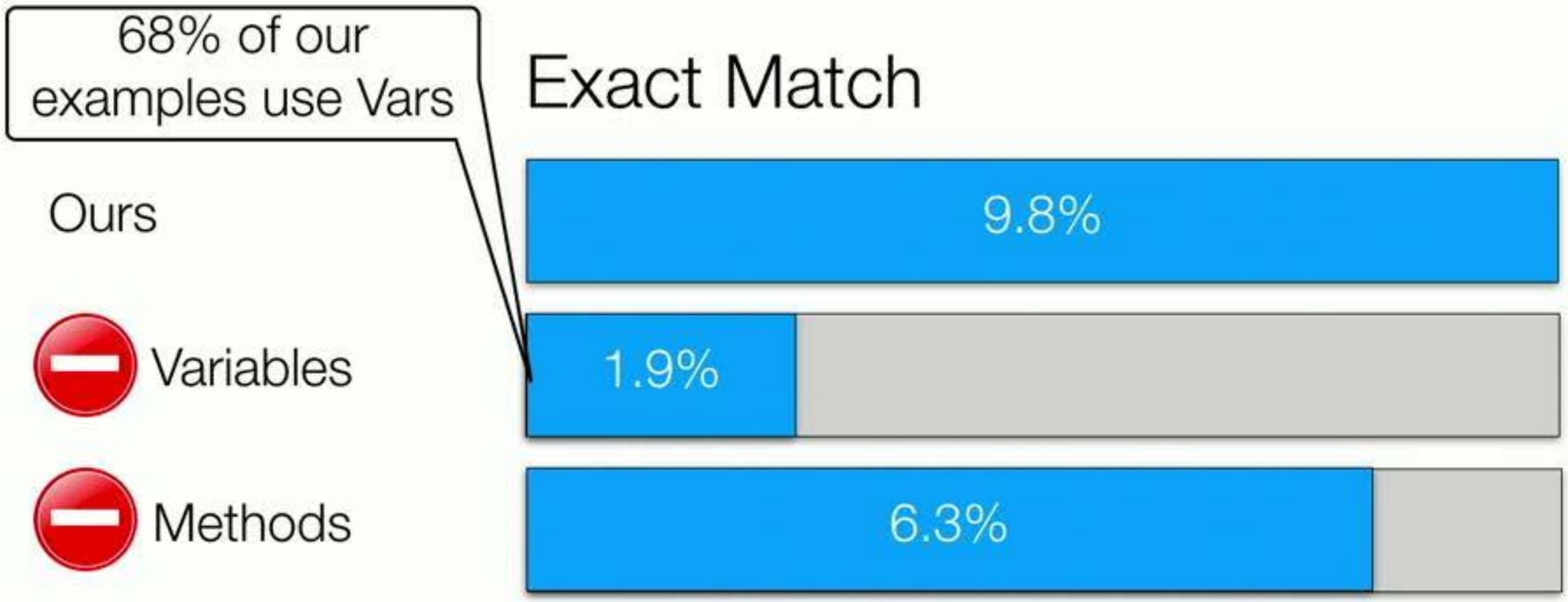
Exact Match and BLEU Score



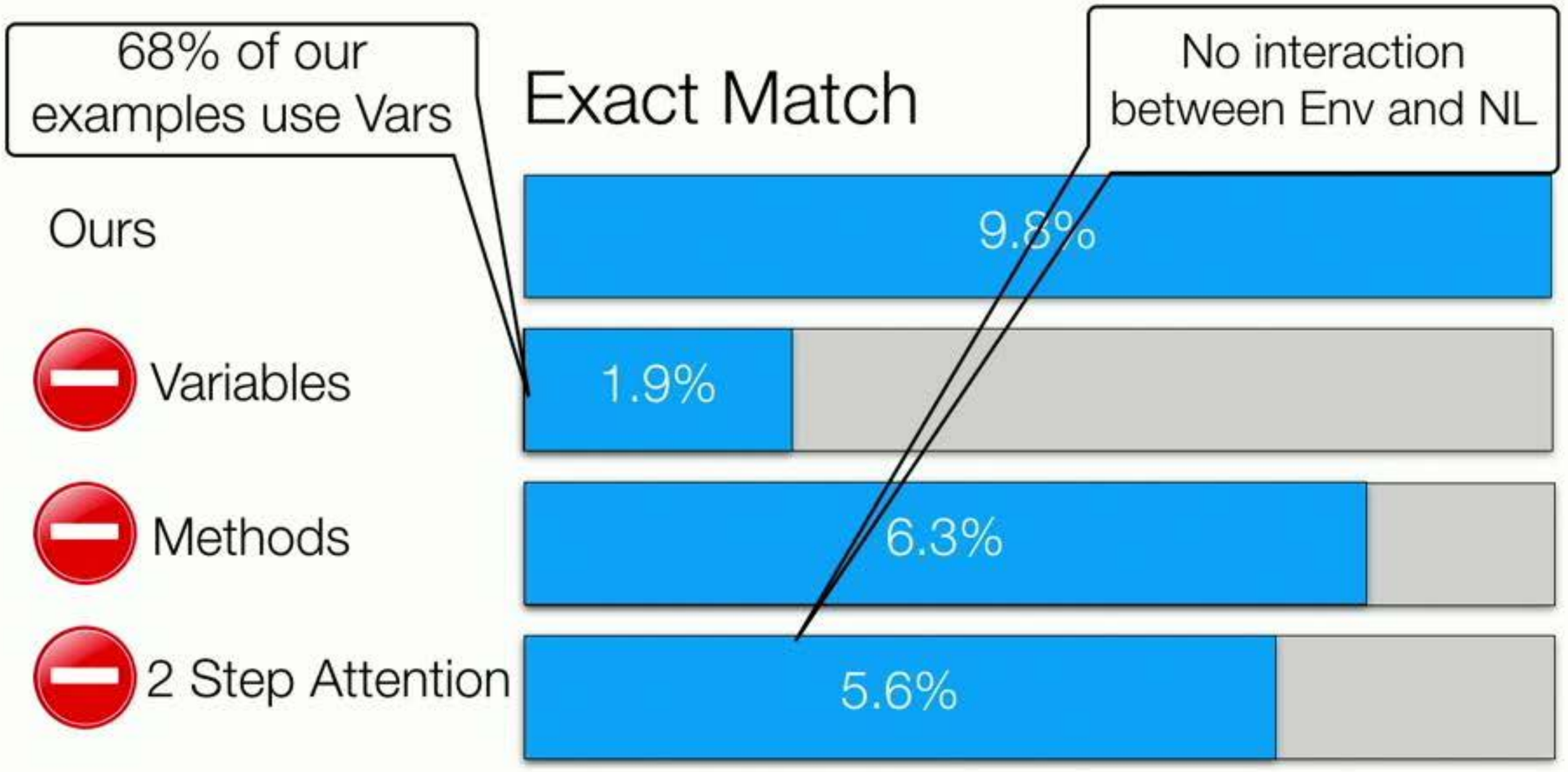
Feature Ablations



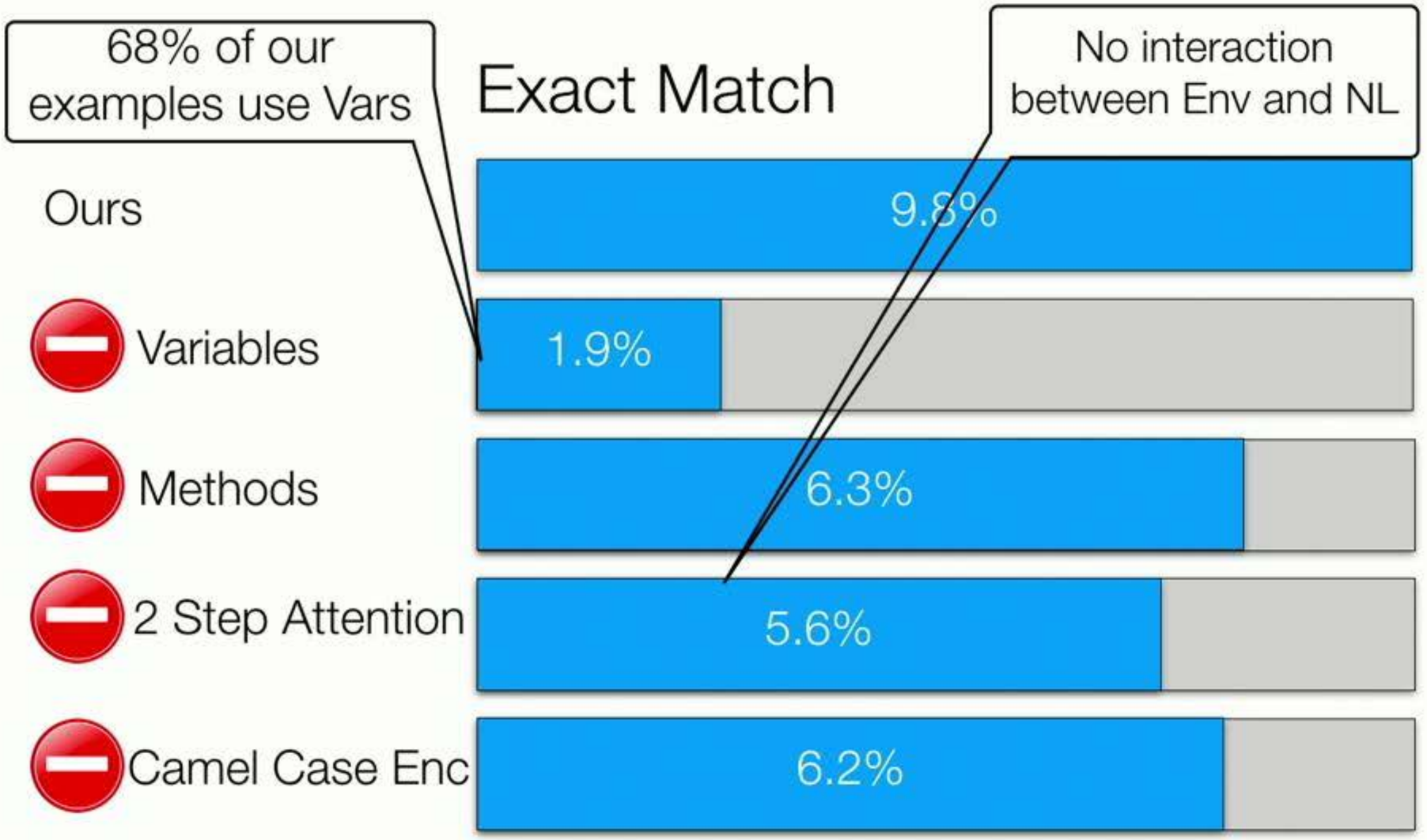
Feature Ablations



Feature Ablations



Feature Ablations



Using **environment variables**

Gets the value of the tags property.
This accessor method returns a
reference to the live list,
not a snapshot.

Variables:

```
String validationPattern;
List<String> tags;
```

Methods:

```
String getValidationPattern
void setValidationPattern
```

Code:

```
List <String> function() {
  if ( tags == null ) {
    tags = new ArrayList <String>();
  }
  return this.tags;
}
```

Calling **user defined methods**

Convert mixed case to underscores.

Variables:

```
NamingStrategy INSTANCE;
```

Methods:

```
String classToTableName
String collectionTableName
String tableName
String columnName
String addUnderscores
```

Code:

```
String function (String arg0) {
  return addUnderscores (arg0);
}
```

More **efficient** ways to write code

Empty the violations list

Variables:

```
long numPptEntries
List<Violation> violations
```

Prediction:

```
void function() {
  violations.clear() ;
}
```

Reference:

```
void function () {
  violations = new
     ArrayList<Violation>();
}
```

Wrong **Types**

Creates and adds a new limit

Variables:

```
List limits
ElementType
  element
```

Methods:

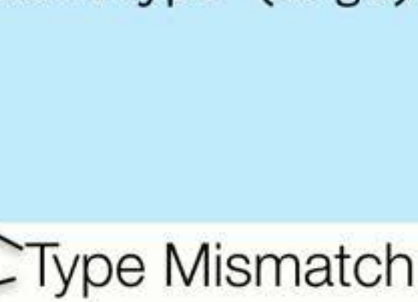
```
List getLimits
ElementType
  getElement
```

Prediction:

```
ElementType function (final ElementType arg0) {
    ElementType loc0 = new ElementType (arg0) ;
    this.limits.add(loc0) ;
    return loc0;
}
```

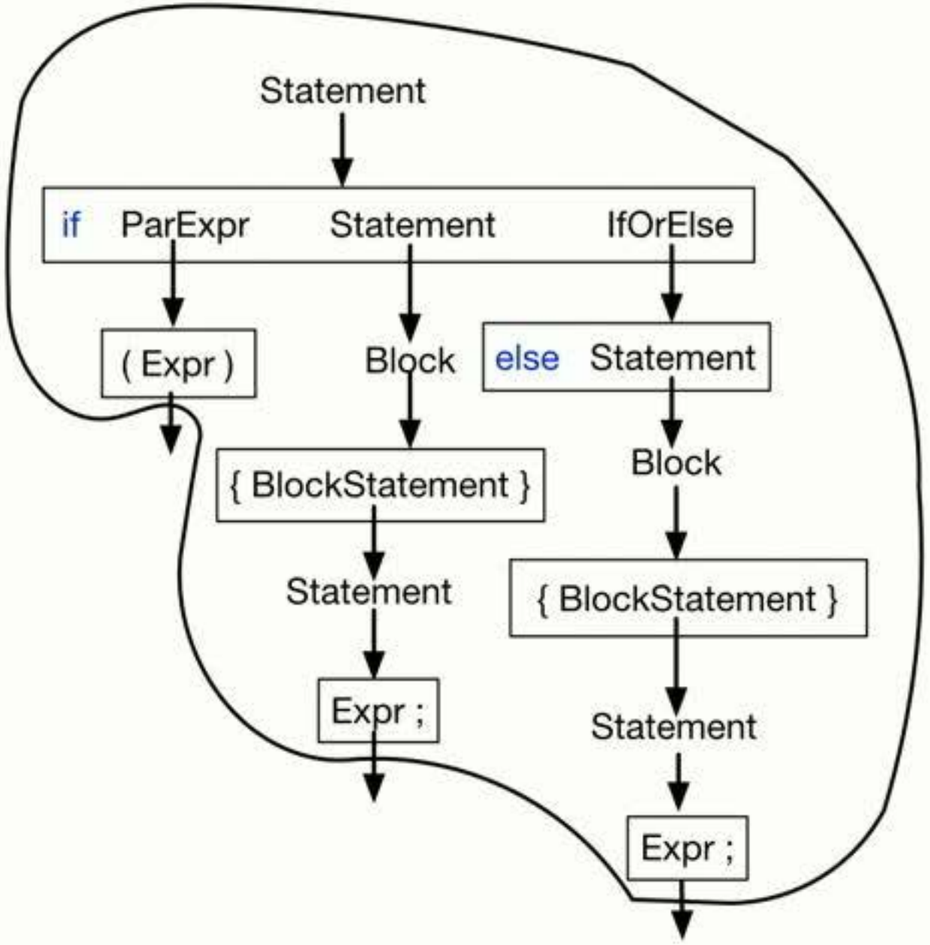
Reference:

```
Limit function () {
    Limit loc0 = new Limit () ;
    this.limits.add(loc0) ;
    return loc0;
}
```



Programmatic Idioms

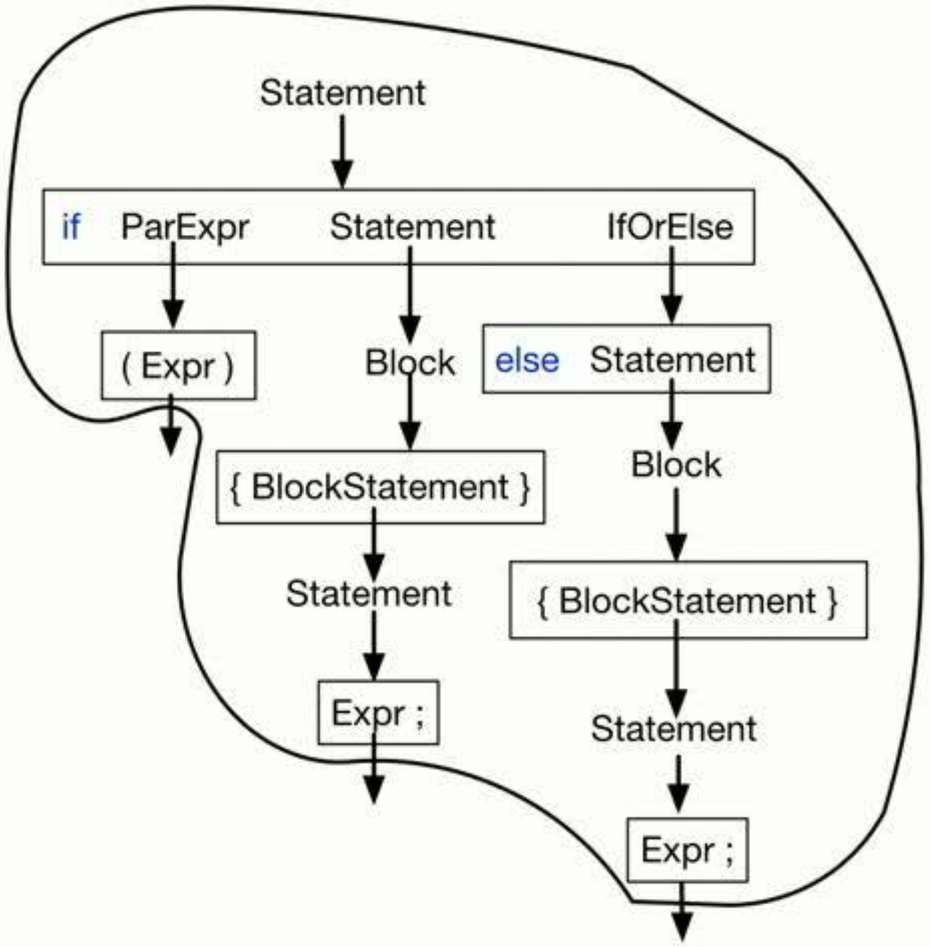
Long Decoding Paths



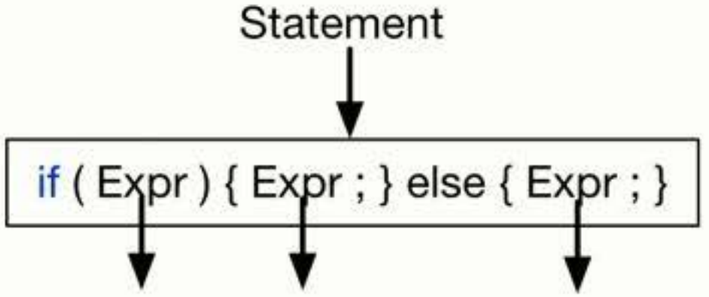
This if-then-else code snippet requires **11 decoding rules** for its structure!

* From Allamanis et al. 2014

Long Decoding Paths

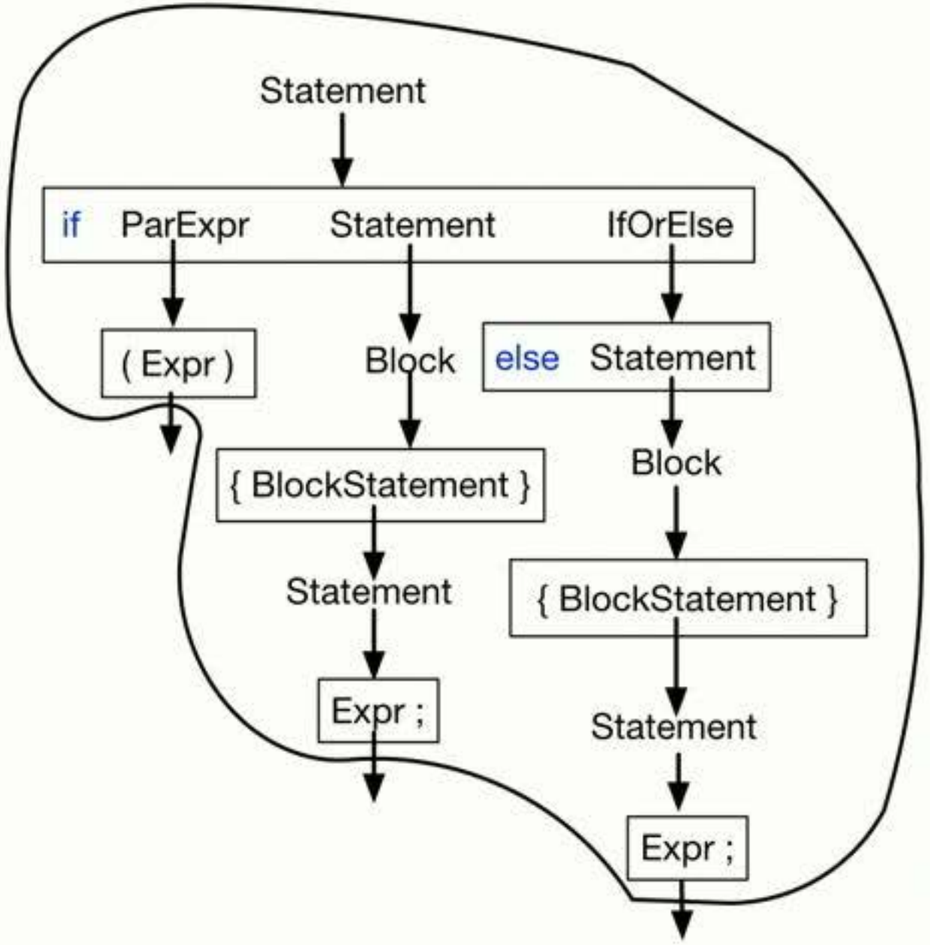


This if-then-else code snippet requires **11 decoding rules** for its structure!

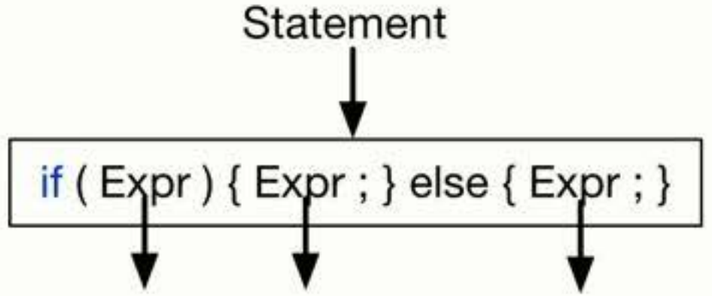


* From Allamanis et al. 2014

Long Decoding Paths



This if-then-else code snippet requires **11 decoding rules** for its structure!



Code Idiom* - A frequently recurring subtree of Program Syntax Trees

* From Allamanis et al. 2014

Programmatic Idioms

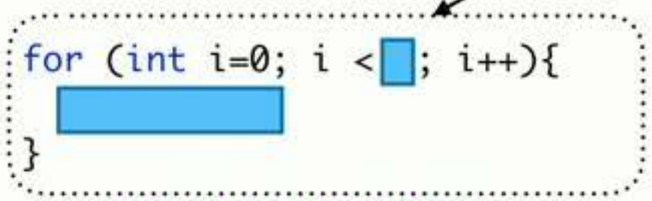
NL Query: Adds a scalar to this vector in place.

```
public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }
```

Programmatic Idioms

NL Query: Adds a scalar to this vector in place.

```
public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }
```

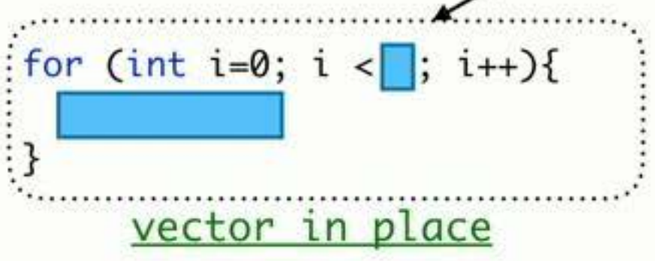


vector in place

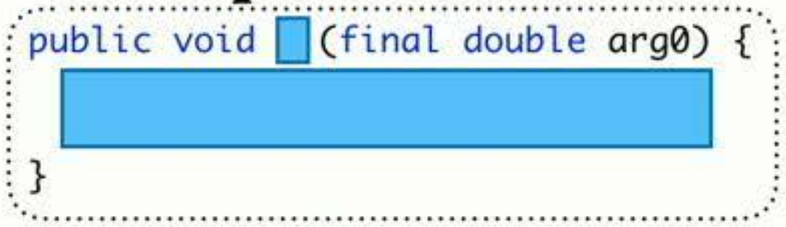
Programmatic Idioms

NL Query: Adds a scalar to this vector in place.

```
public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }
```



vector in place

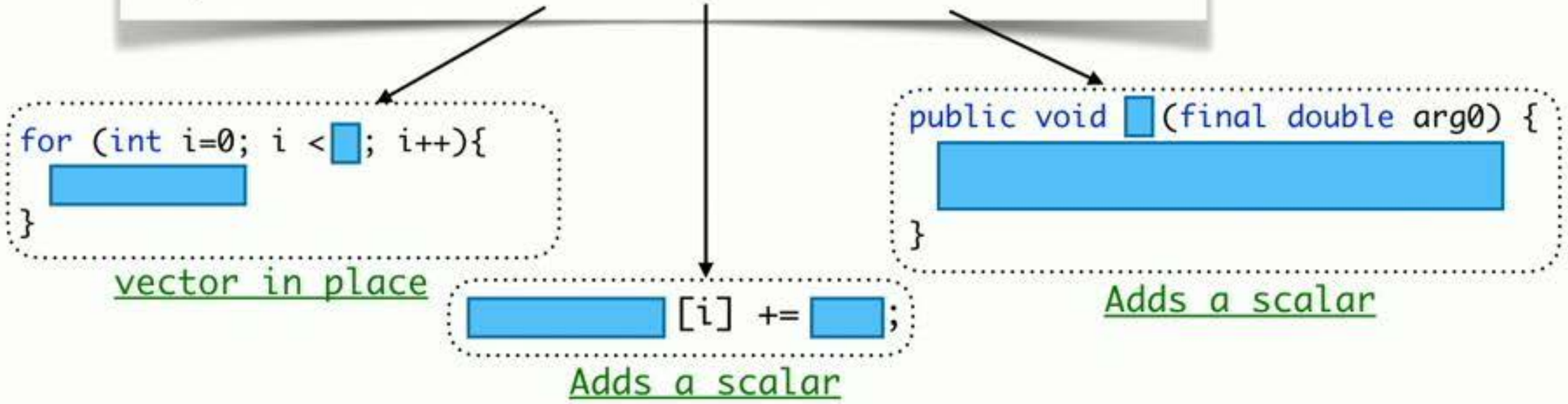


Adds a scalar

Programmatic Idioms

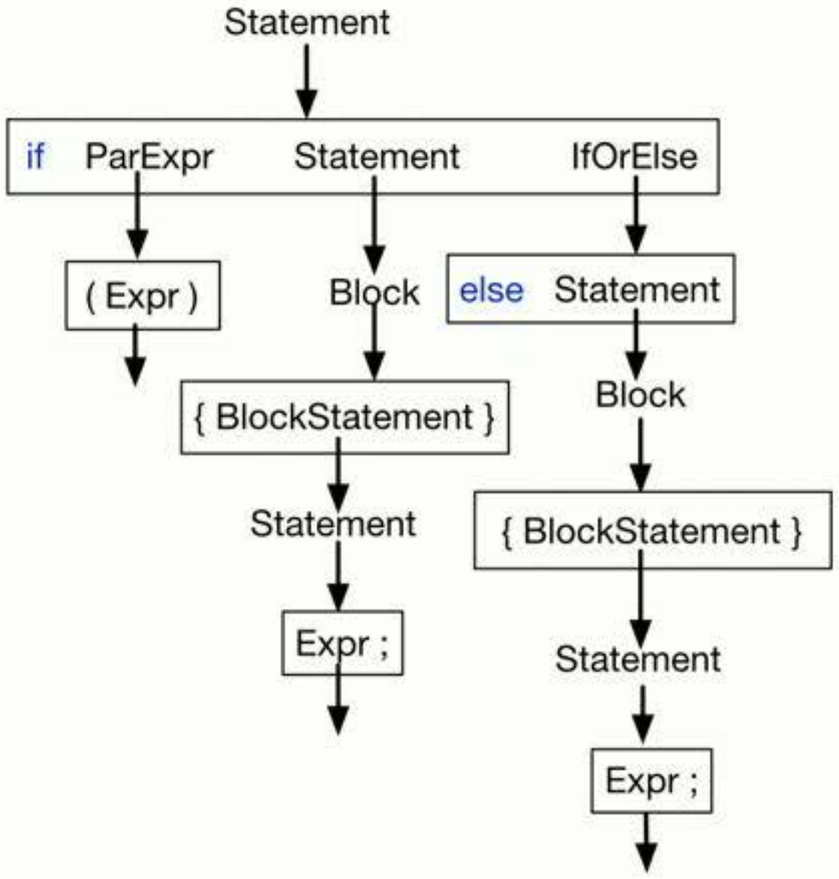
NL Query: Adds a scalar to this vector in place.

```
public void add(final double arg0) {
    for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
     }
  }
```

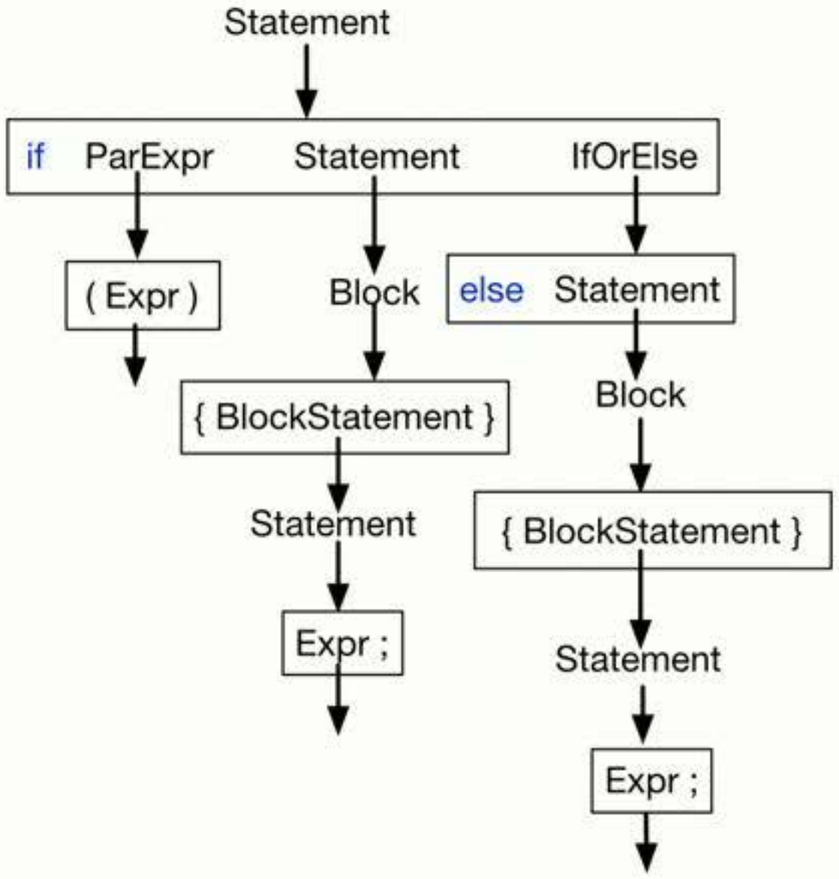


Idiom Extraction

Idiom Extraction

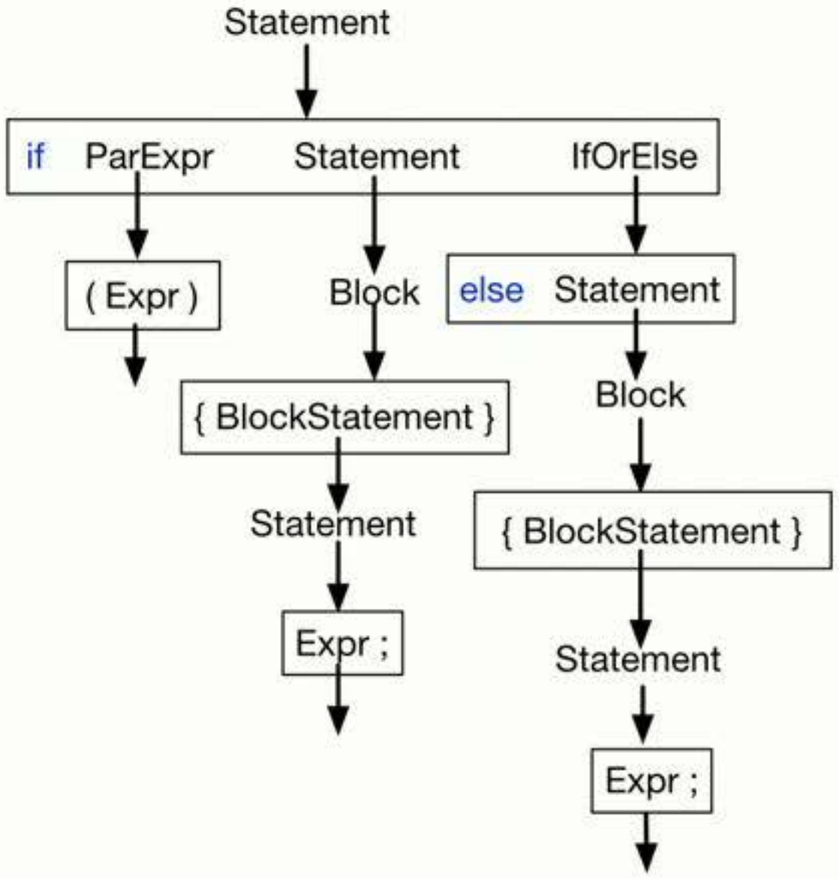


Idiom Extraction



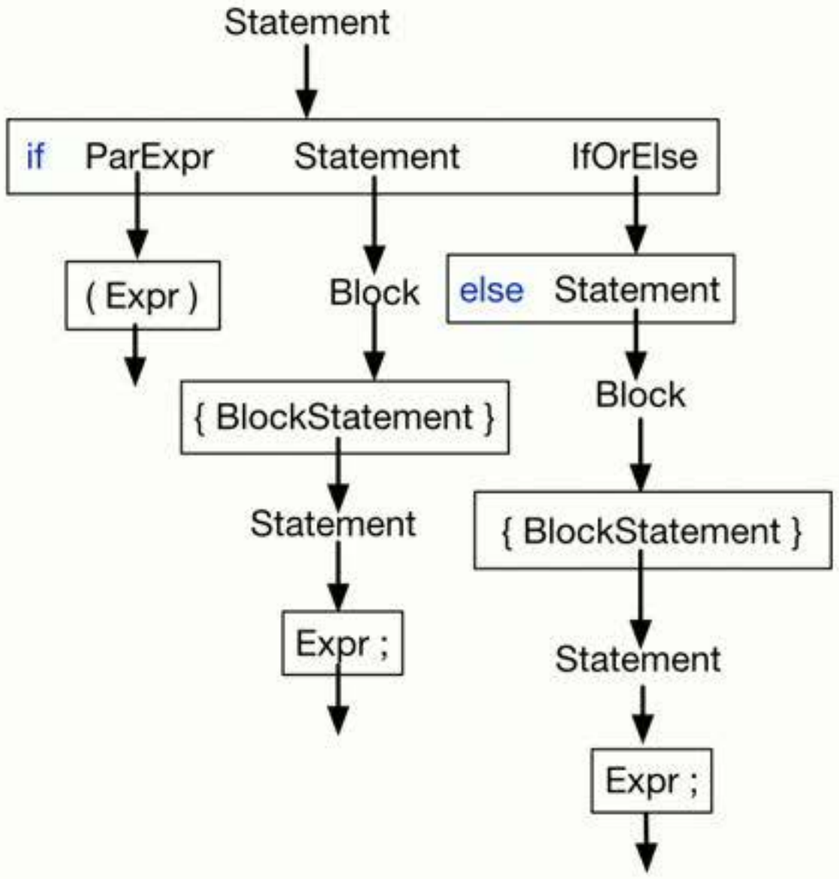
- Use top-K most frequent subtrees from a large code corpus

Idiom Extraction



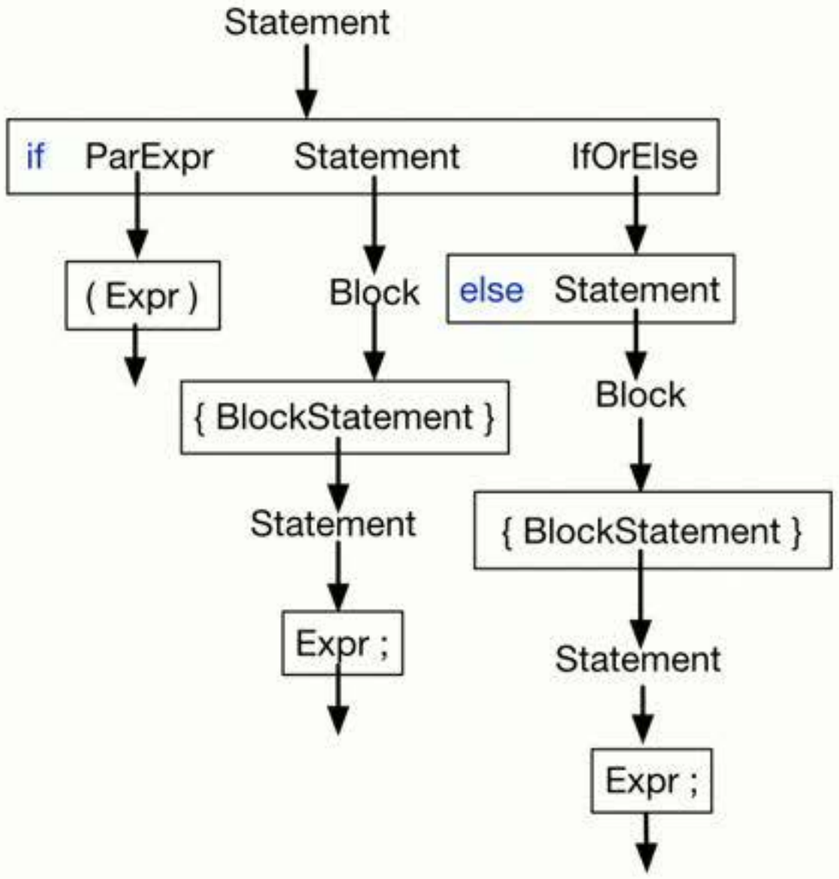
- Use top-K most frequent subtrees from a large code corpus
- **Exponentially** many subtrees to enumerate!

Idiom Extraction



- Use top-K most frequent subtrees from a large code corpus
- **Exponentially** many subtrees to enumerate!
- If s' subtree-of s, then freq(s') ≥ freq(s)

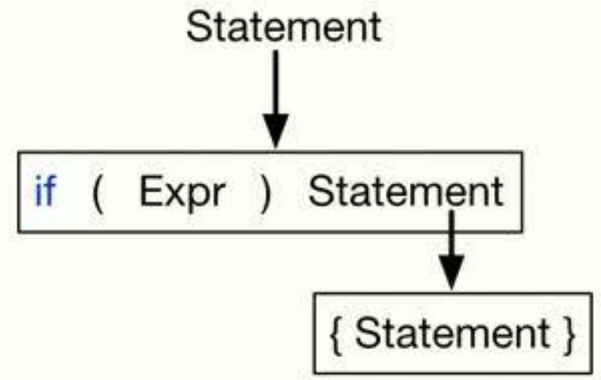
Idiom Extraction



- Use top-K most frequent subtrees from a large code corpus
- **Exponentially** many subtrees to enumerate!
- If s' subtree-of s, then freq(s') ≥ freq(s)
- Solution: Repeatedly **merge frequently occurring depth-2 subtrees** like BPE

Merge depth-2 subtrees

- Step 1: Find the **most frequent depth-2 subtree** (t)

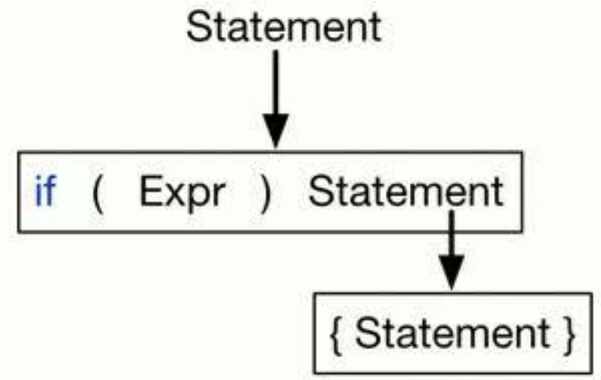


Statement → if (Expr) Statement

Statement → { return Expr ; }

Merge depth-2 subtrees

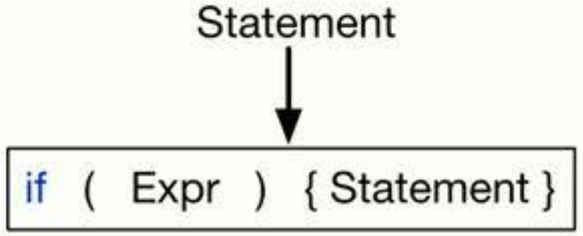
- Step 1: Find the **most frequent depth-2 subtree** (t)



Statement → if (Expr) Statement

Statement → { return Expr ; }

- Step 2: **Collapse** the subtree into an idiom

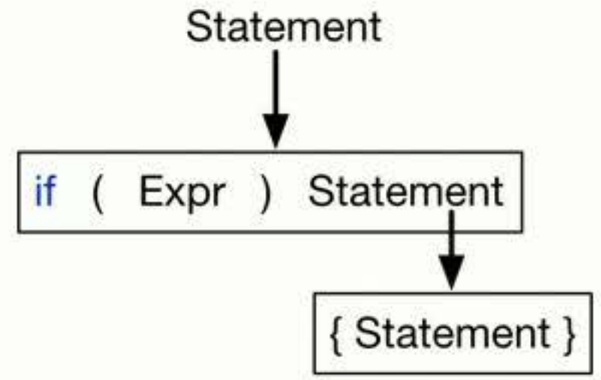


New Idiom Rule:

Statement → if (Expr) { Statement }

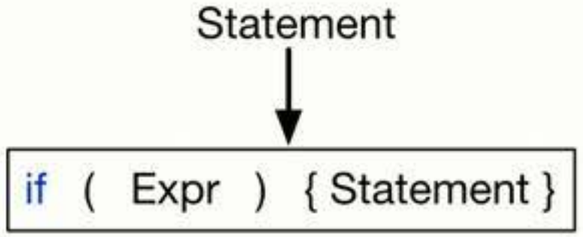
Merge depth-2 subtrees

- Step 1: Find the **most frequent depth-2 subtree** (t)



Statement → if (Expr) Statement
Statement → { return Expr ; }

- Step 2: **Collapse** the subtree into an idiom

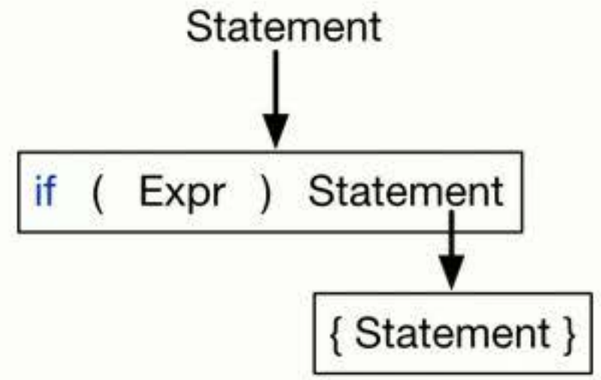


New Idiom Rule:
Statement → if (Expr) { Statement }

- Step 3: **Replace** all occurrences of t with the idiom

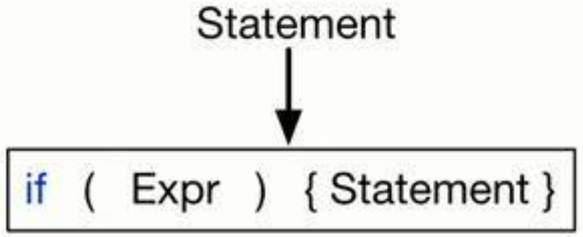
Merge depth-2 subtrees

- Step 1: Find the **most frequent depth-2 subtree** (t)



Statement → if (Expr) Statement
Statement → { return Expr ; }

- Step 2: **Collapse** the subtree into an idiom

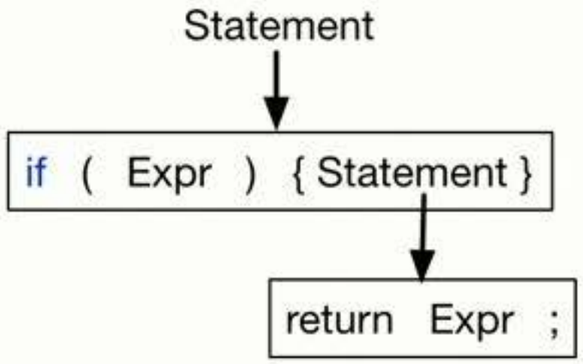


New Idiom Rule:
Statement → if (Expr) { Statement }

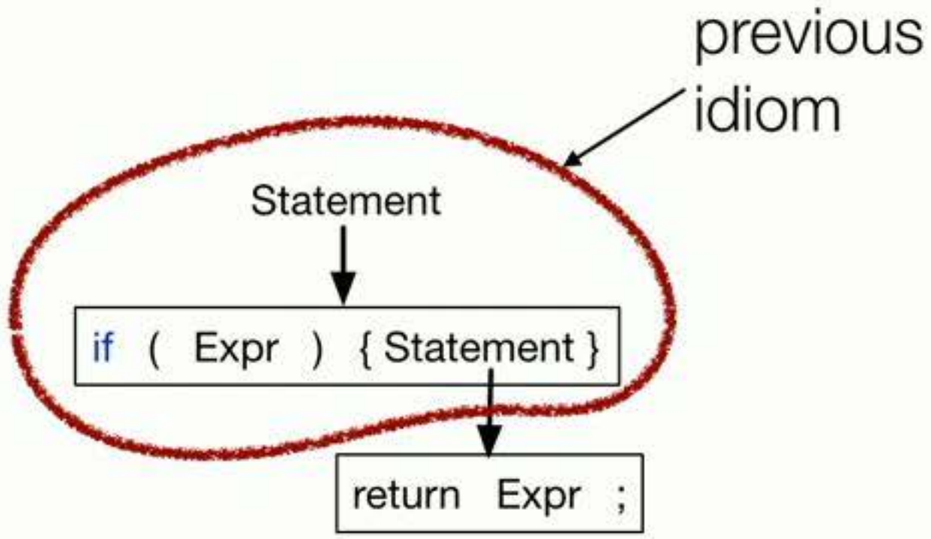
- Step 3: **Replace** all occurrences of t with the idiom

- Step 4: **Repeat,** to get Idiom set, $\mathcal{I}$

Depth-k Idioms

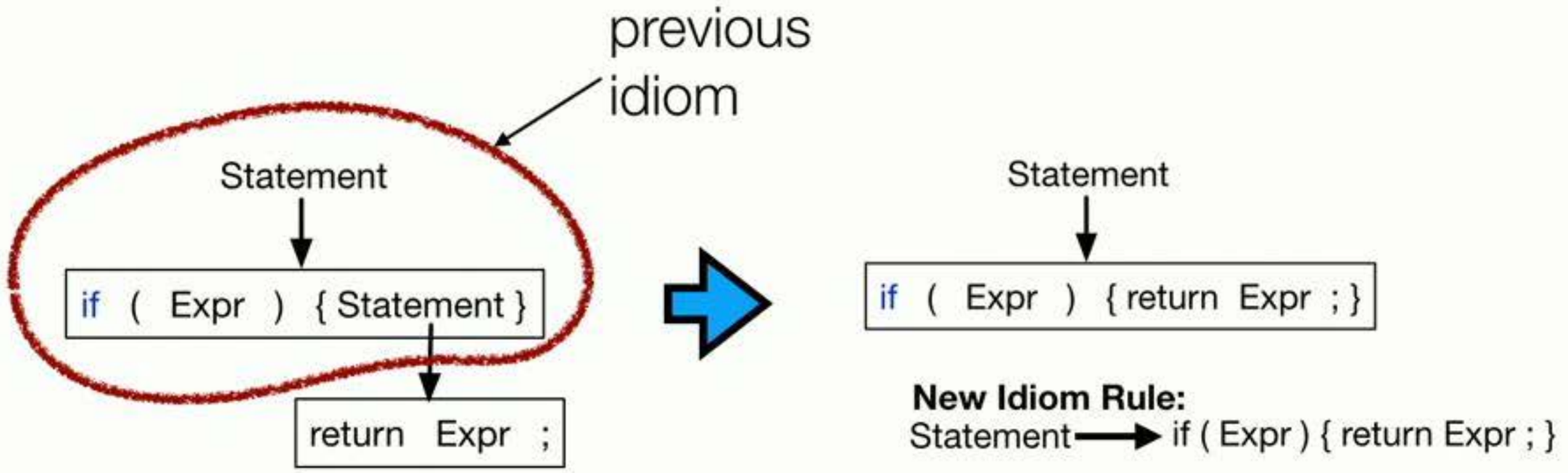


Depth-k Idioms



Depth-2 subtree
contains **previously**
created idiom

Depth-k Idioms



Depth-2 subtree contains **previously** created idiom

Depth-3 idiom

Idiom-Aware Decoding

Idiom Application

- **Idea:** Add Idioms as new **parsing rules** in decoder grammar ($\mathcal{R} \cup \mathcal{I}$).

Idiom Application

- **Idea:** Add Idioms as new **parsing rules** in decoder grammar ($\mathcal{R} \cup \mathcal{I}$).
- Use shortest parse for training.

Idiom Application

- **Idea:** Add Idioms as new **parsing rules** in decoder grammar $(\mathcal{R} \cup \mathcal{I})$.
- Use shortest parse for training.
- **Implementation:** Convert training set into **Idiom-Compressed syntax trees.**

Idiom Application

- **Idea:** Add Idioms as new **parsing rules** in decoder grammar $(\mathcal{R} \cup \mathcal{I})$.
- Use shortest parse for training.
- **Implementation:** Convert training set into **Idiom-Compressed syntax trees.**
- Let decoder extract parsing rules automatically.

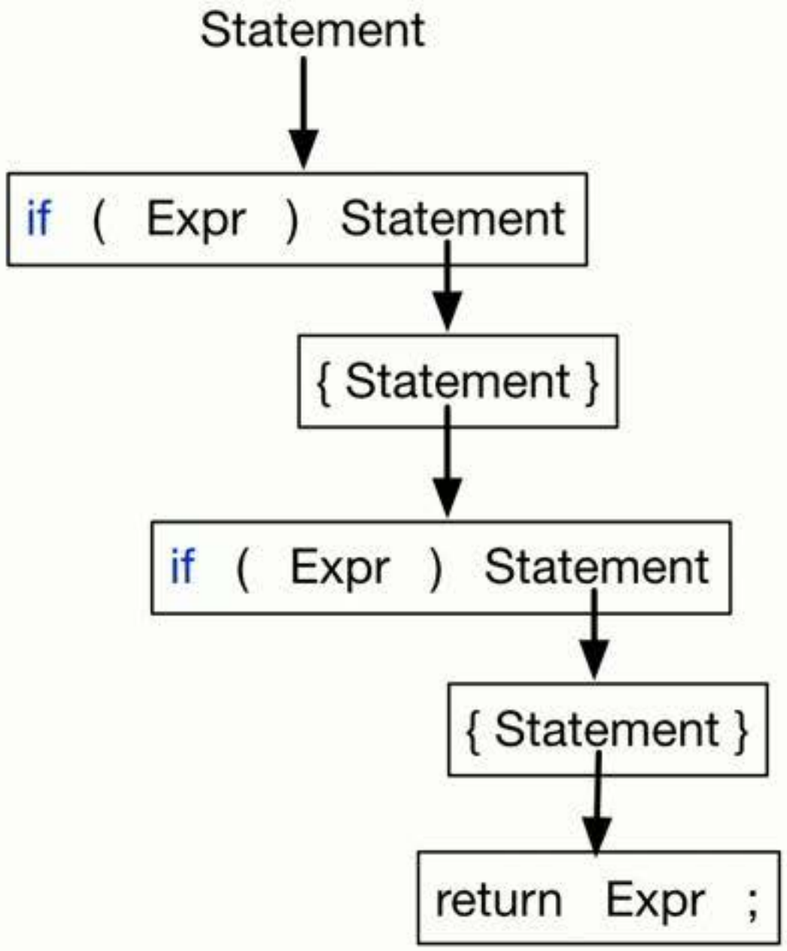
Idiom Application

Apply all idioms in order of extraction to every training example

Procedure **Compress**:

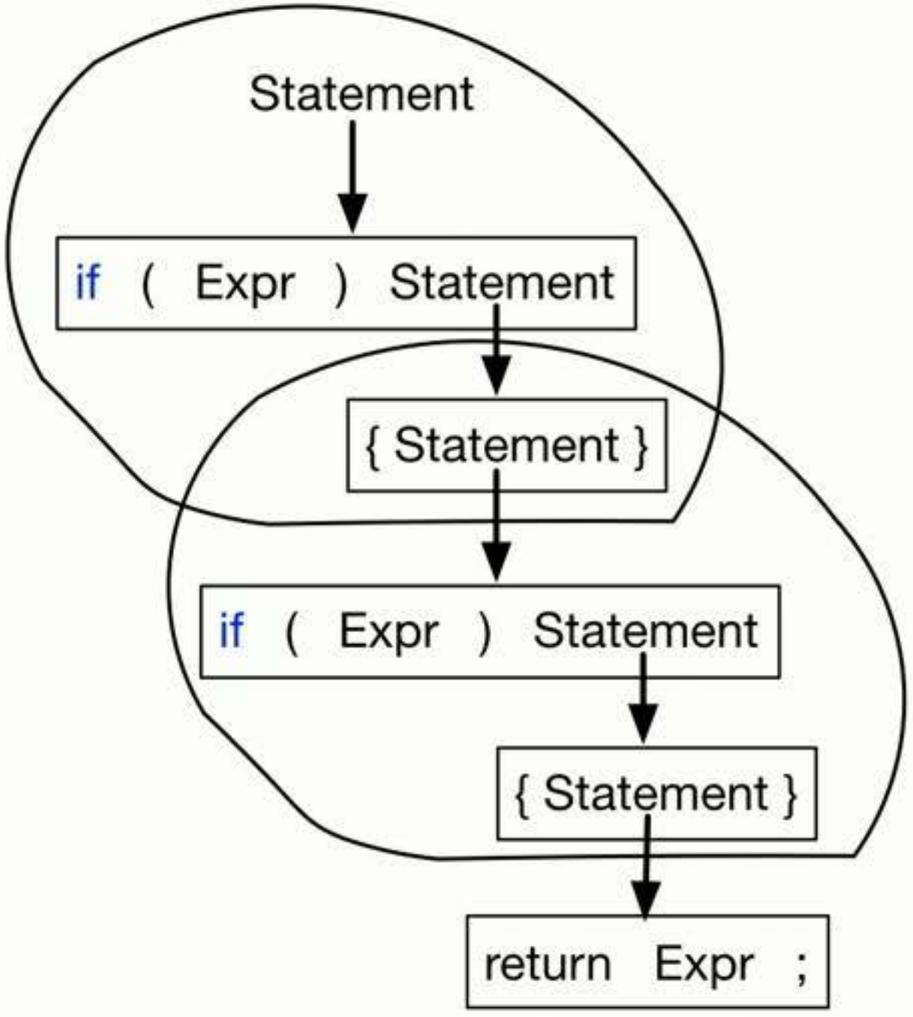
- For c in Training-Set:
  - ***t*** ← **Syntax-Tree**(c)
  - For ***i*** in Idioms:
    - ➡ ***t*** ← **Apply**(***t***, ***i***)

Idiom-Compressed Syntax Trees



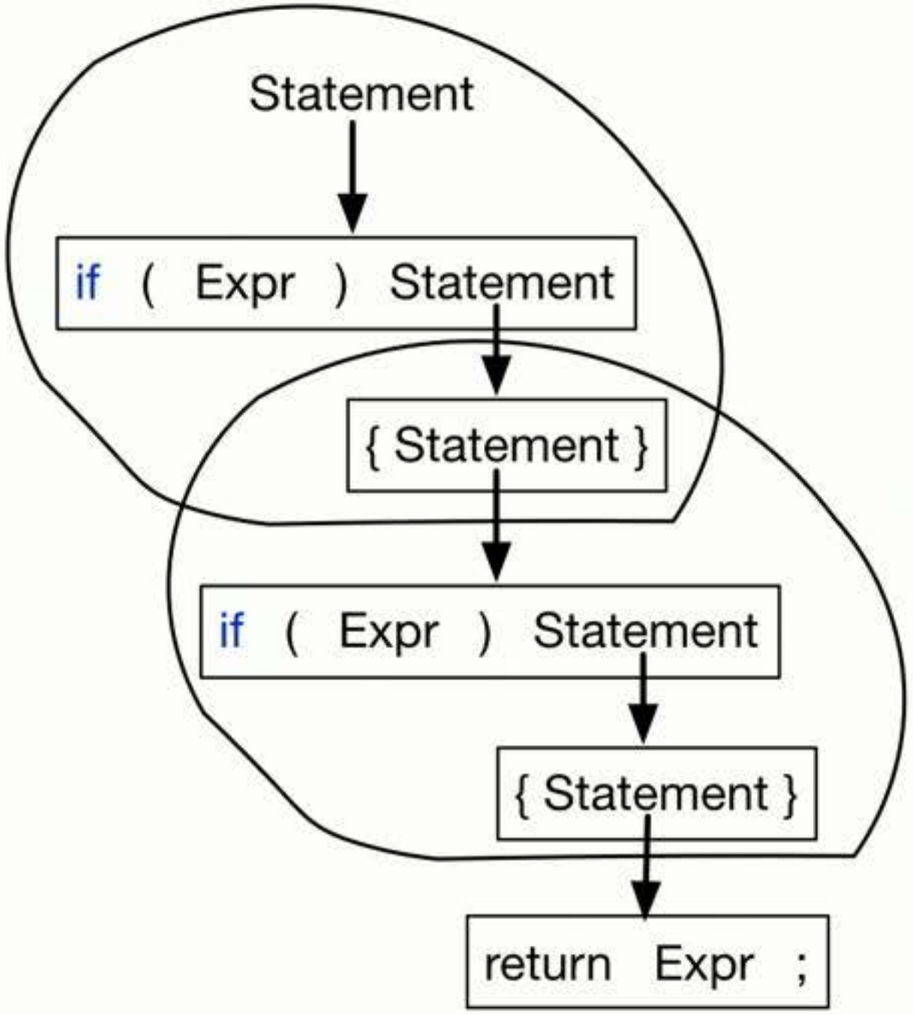
Training Example

Idiom-Compressed Syntax Trees

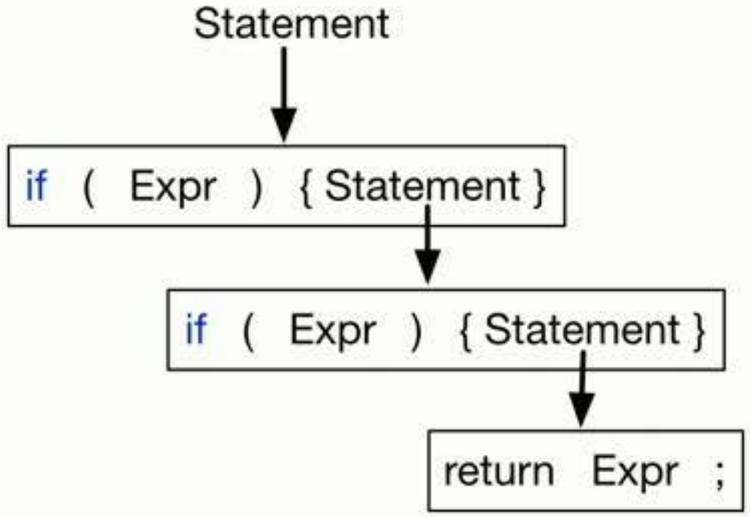


Training Example

Idiom-Compressed Syntax Trees

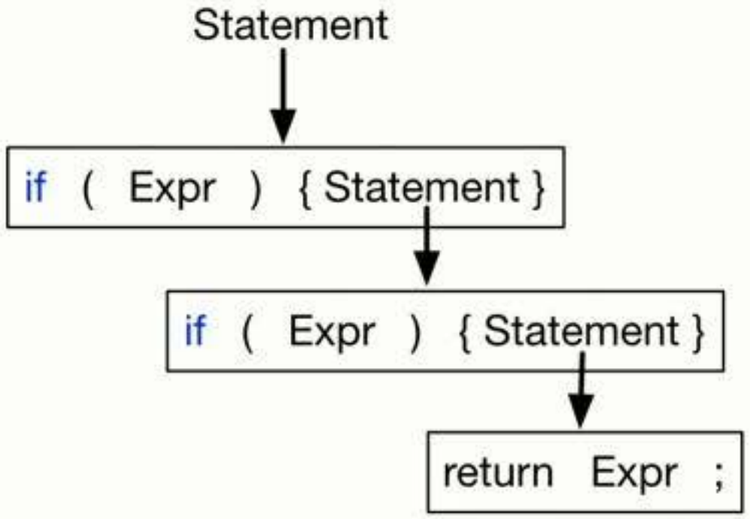


Training Example



Two applications of the idiom
Statement → if (Expr) { Statement }

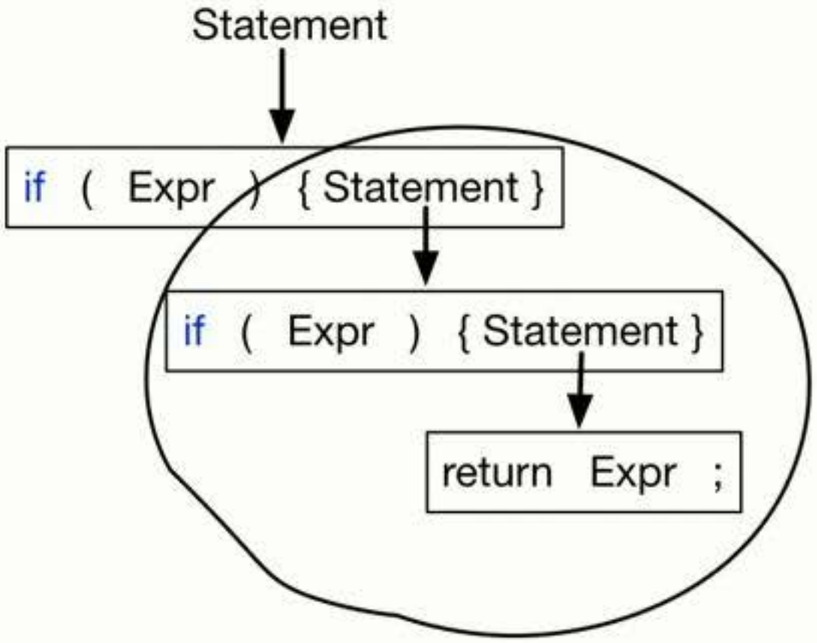
Idiom-Compressed Syntax Trees



Grammar Rules from Compressed Syntax Tree**:**

Statement → if (Expr) { Statement }
Statement → if (Expr) { return Expr; }

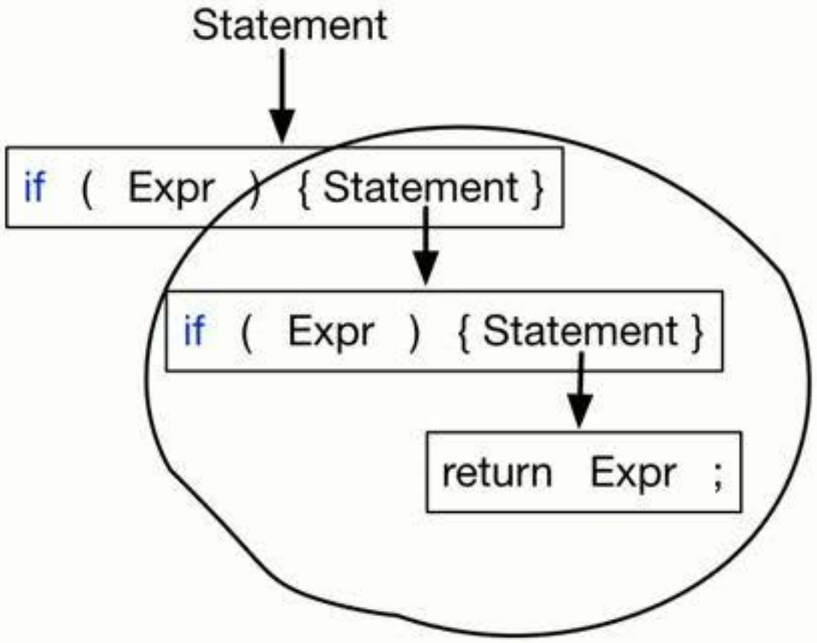
Idiom-Compressed Syntax Trees

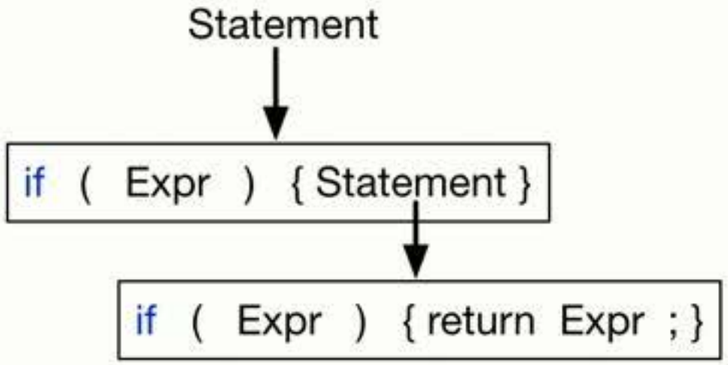


Grammar Rules from Compressed Syntax Tree**:**

Statement → if (Expr) { Statement }
Statement → if (Expr) { return Expr; }

Idiom-Compressed Syntax Trees





One application of the idiom

Statement → if (Expr) { return Expr ; }

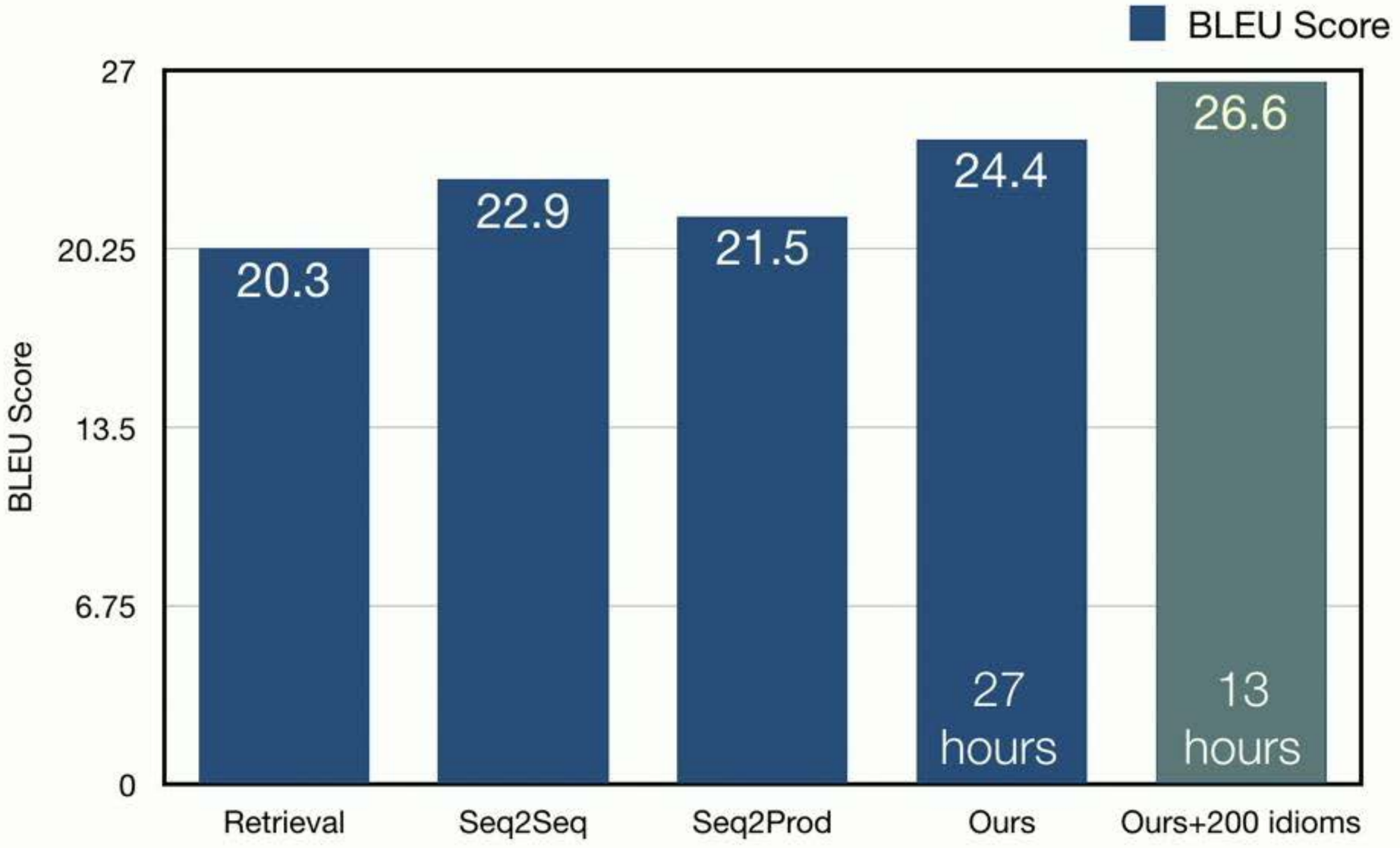
Grammar Rules from Compressed Syntax Tree**:**

Statement → if (Expr) { Statement }

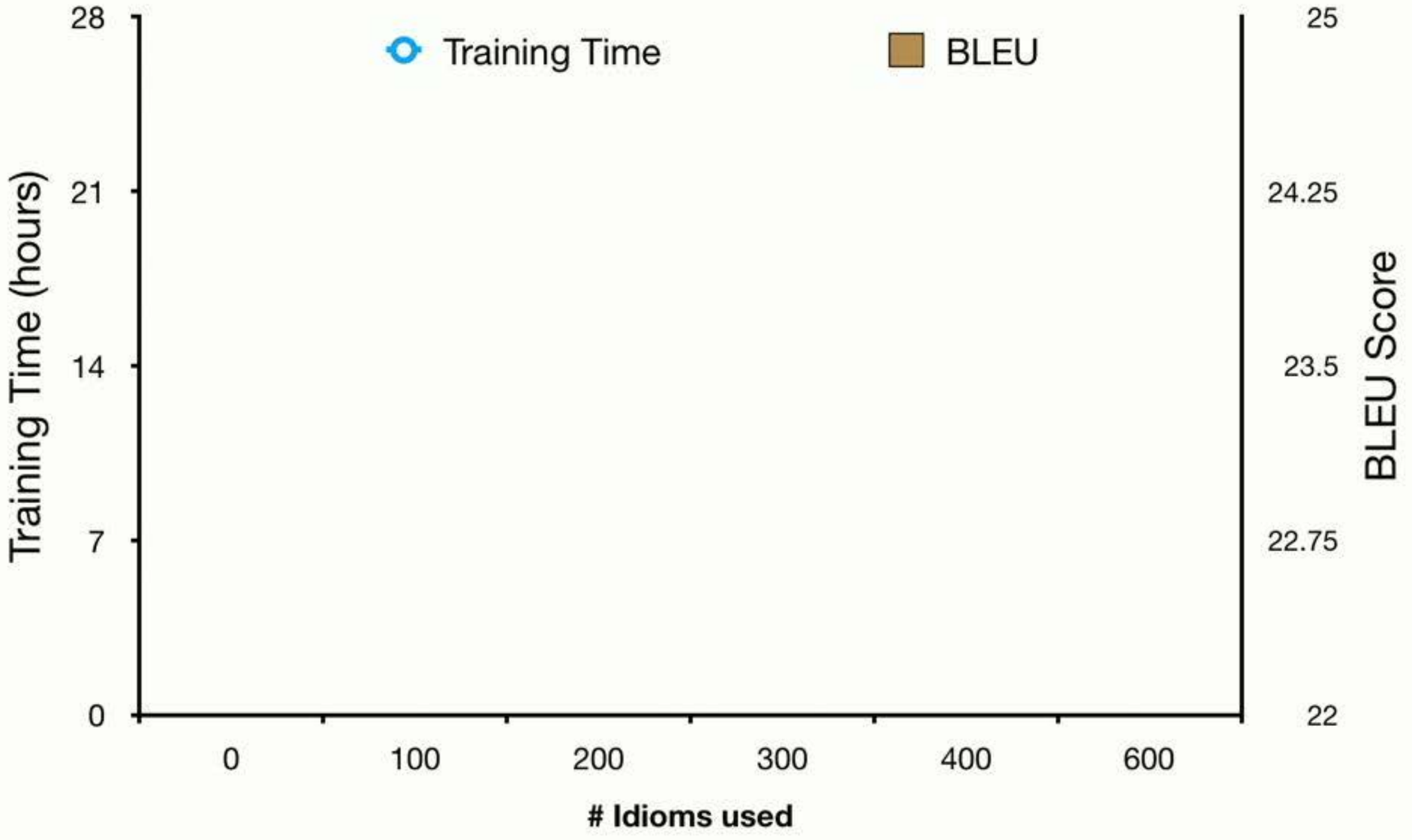
Statement → if (Expr) { return Expr; }

CONCODE using 200 Idioms

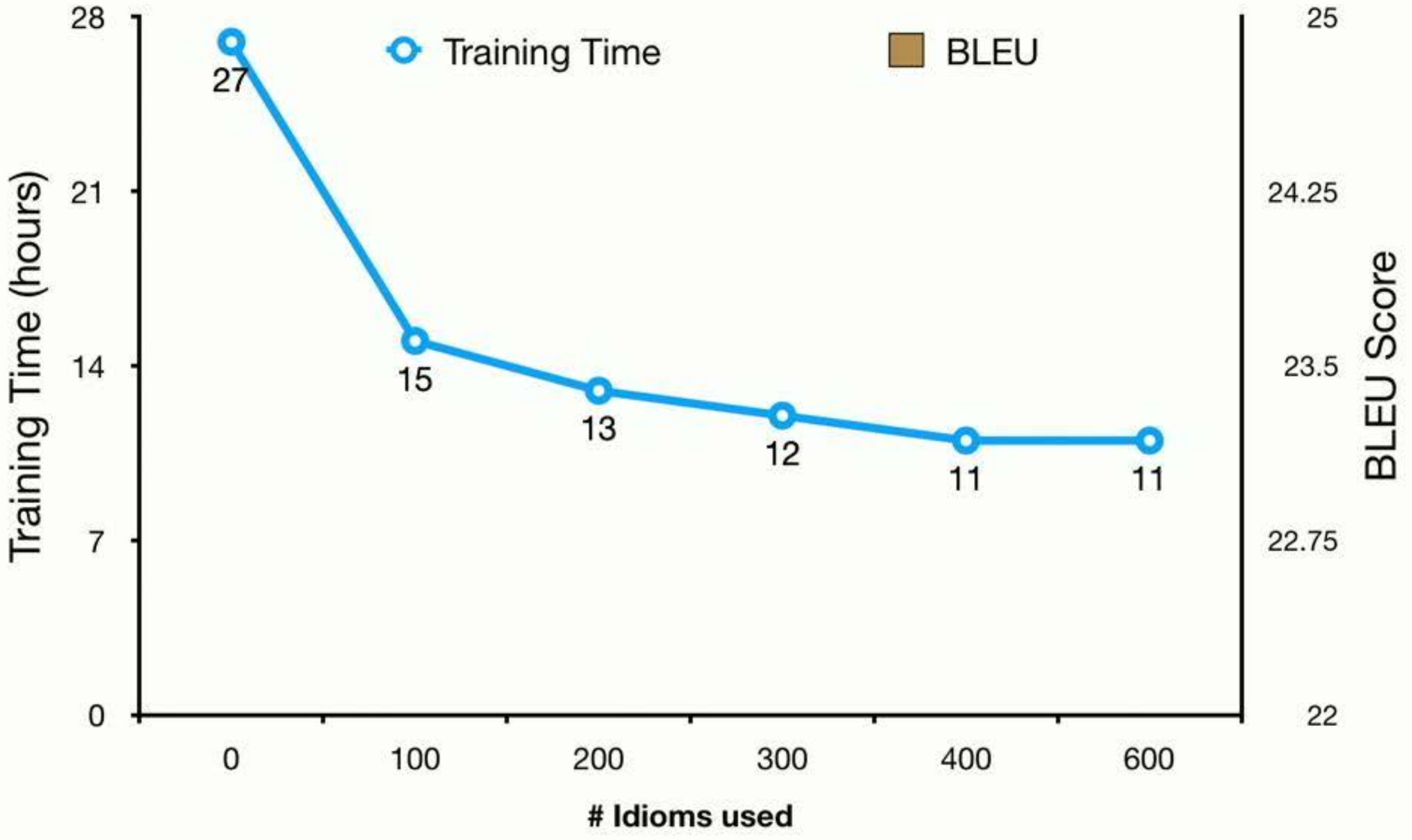
CONCODE using 200 Idioms



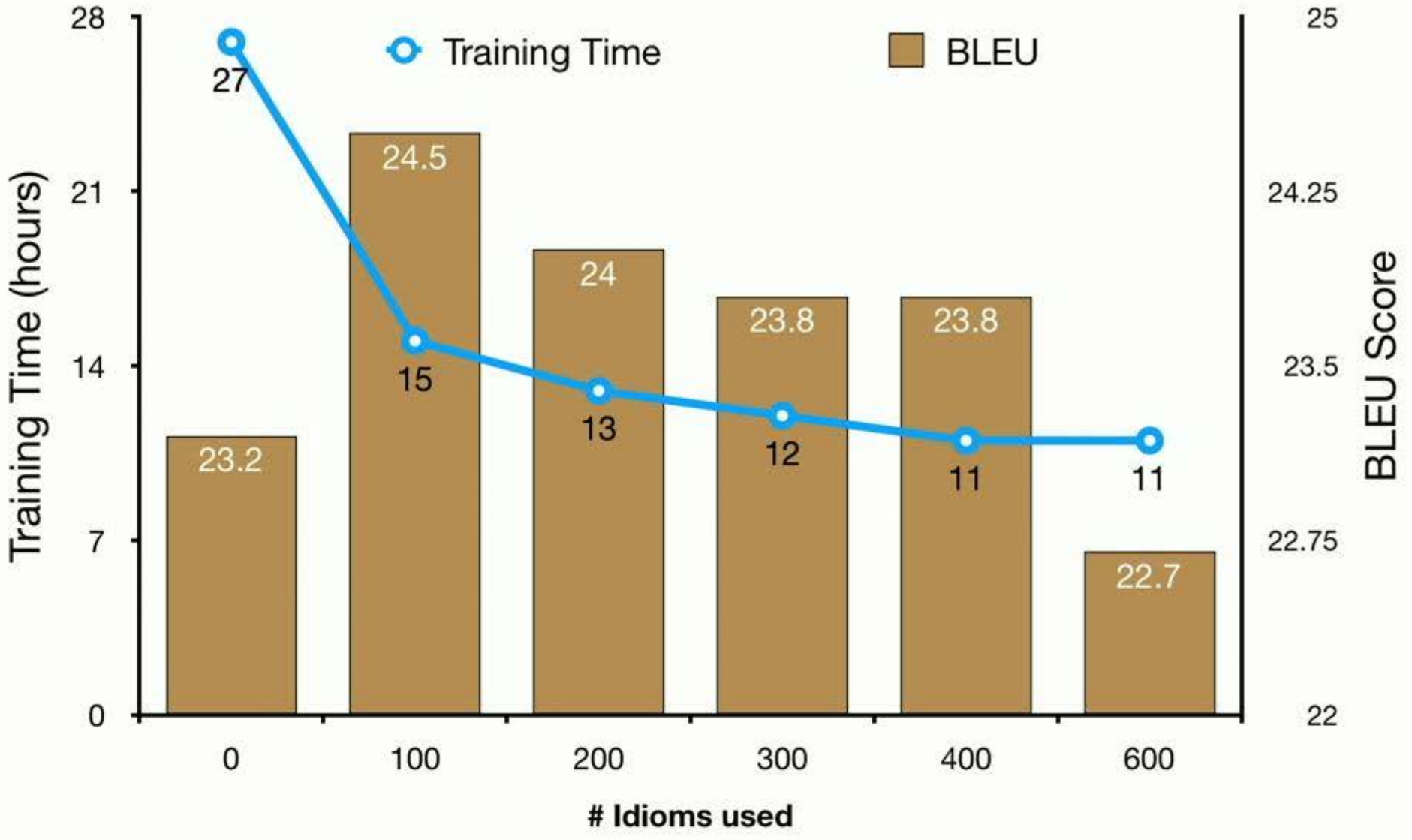
Training Time/Accuracy vs # Idioms



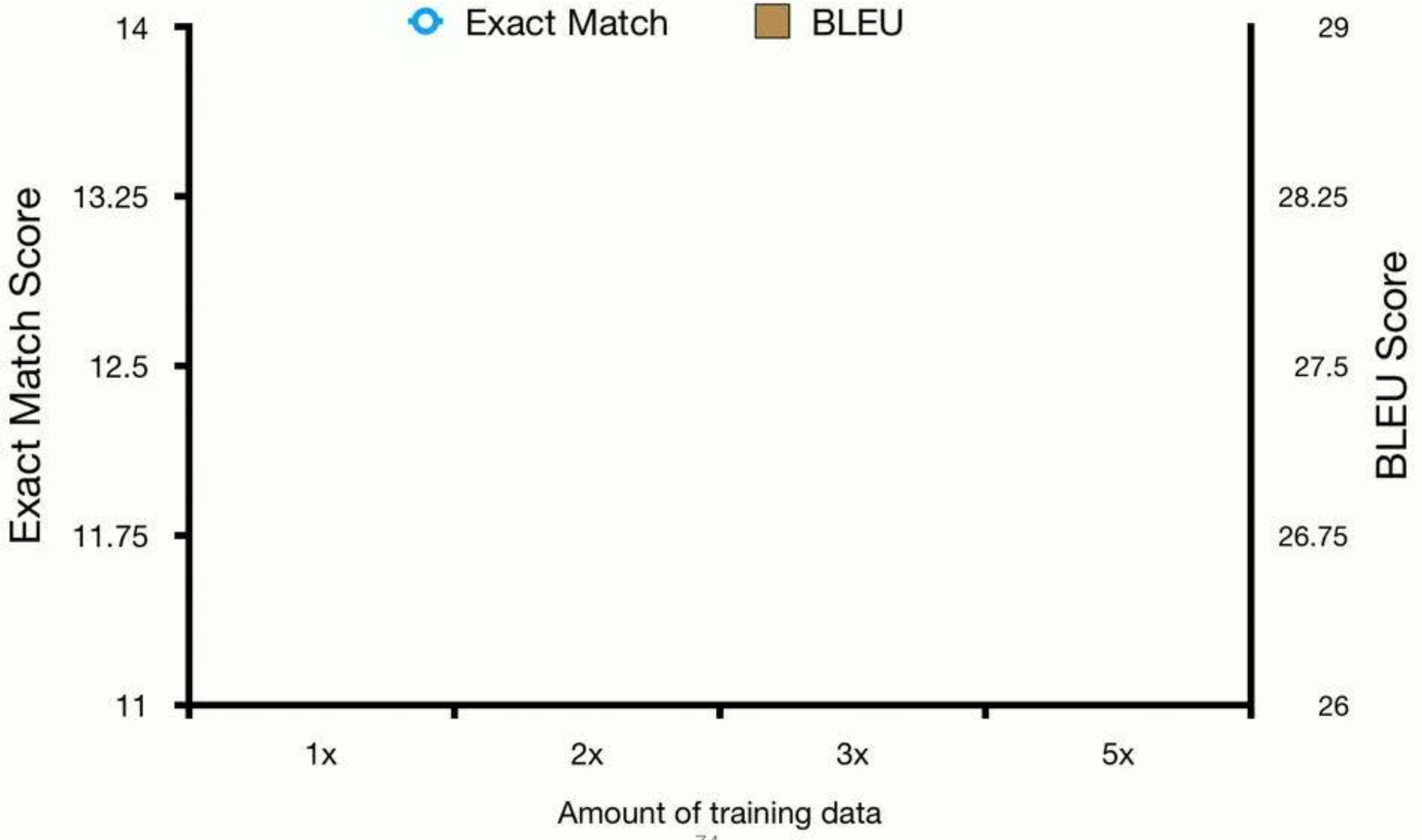
Training Time/Accuracy vs # Idioms



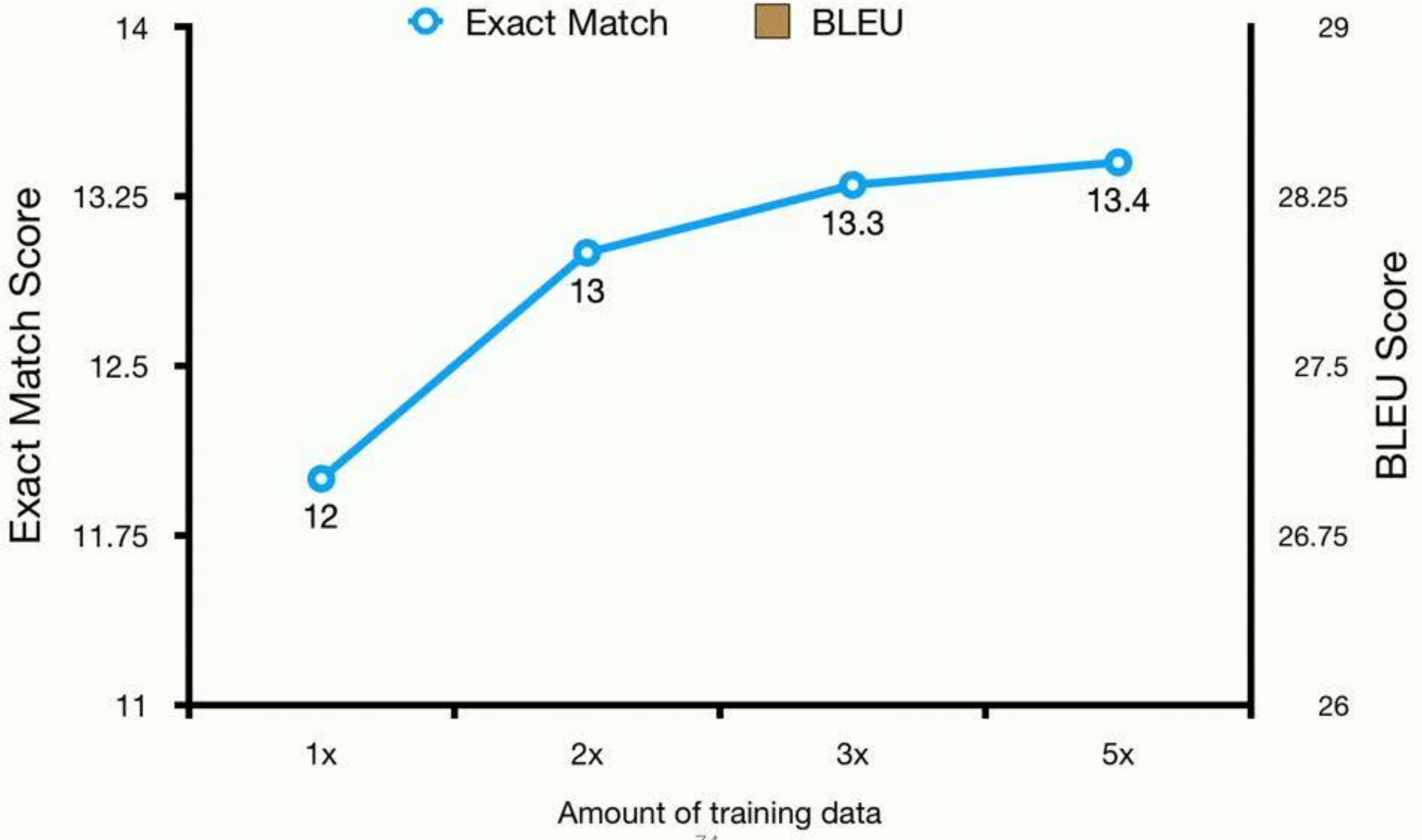
Training Time/Accuracy vs # Idioms



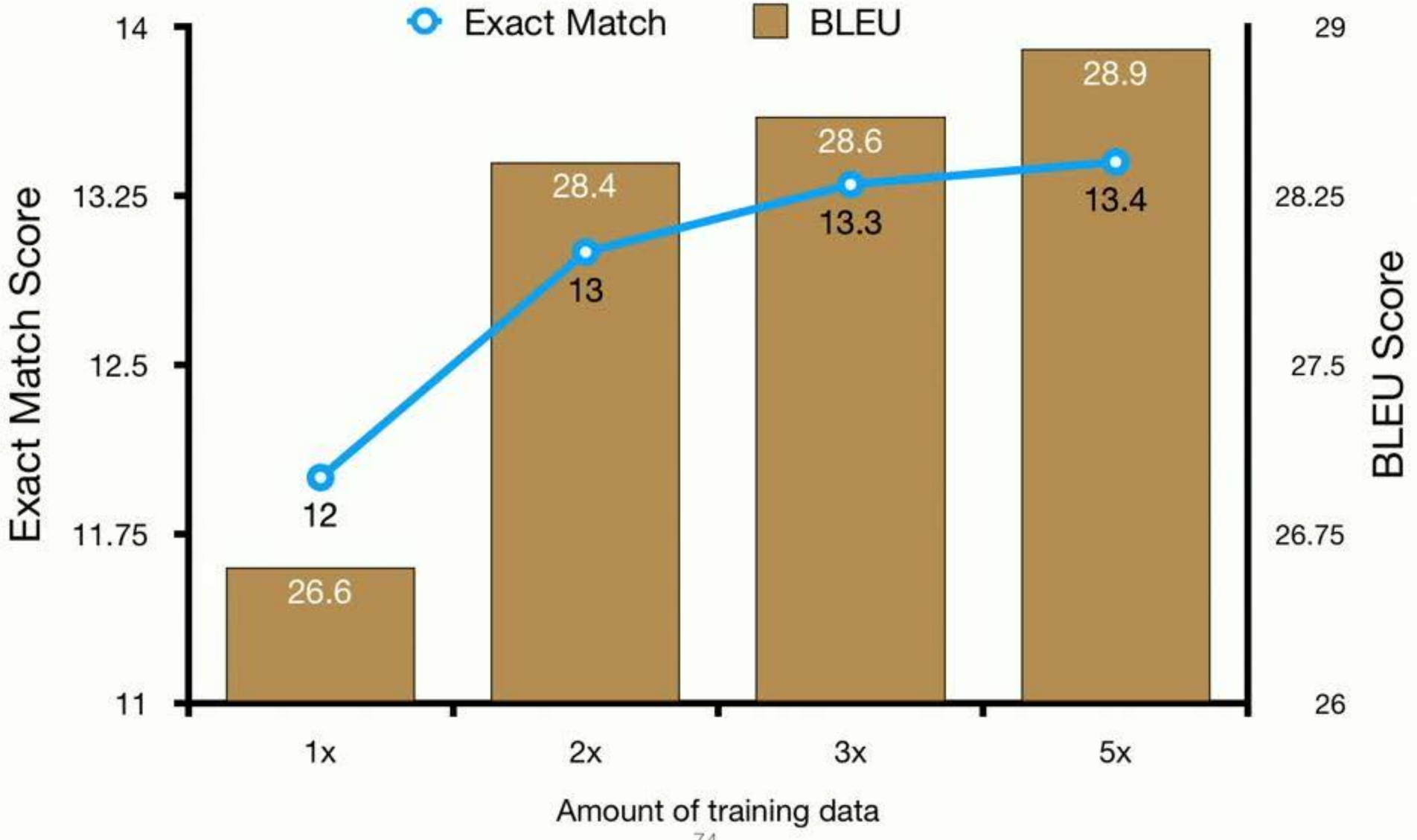
CONCODE + 400 idioms on more data!



CONCODE + 400 idioms on more data!



CONCODE + 400 idioms on more data!

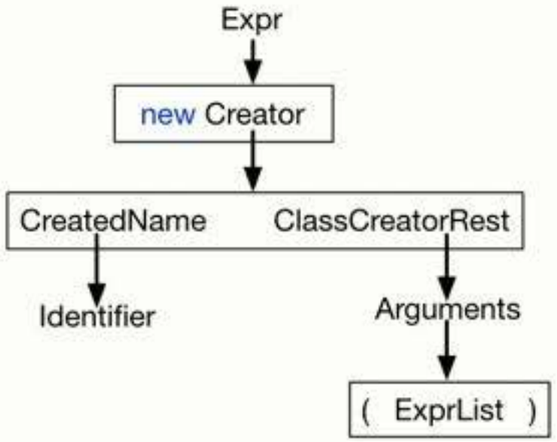


Idioms from CONCODE

Idioms from CONCODE

Instantiation of a
new object

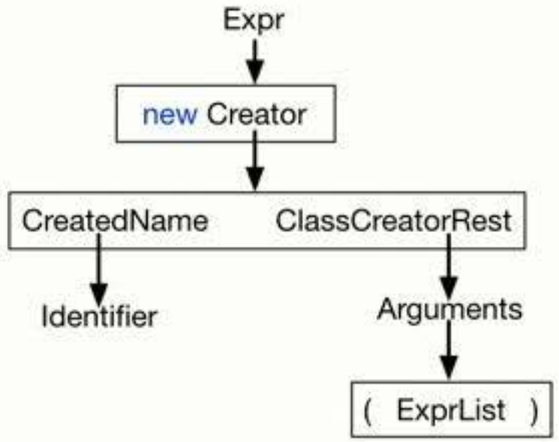
Expr → new Identifier (ExprList)



Idioms from CONCODE

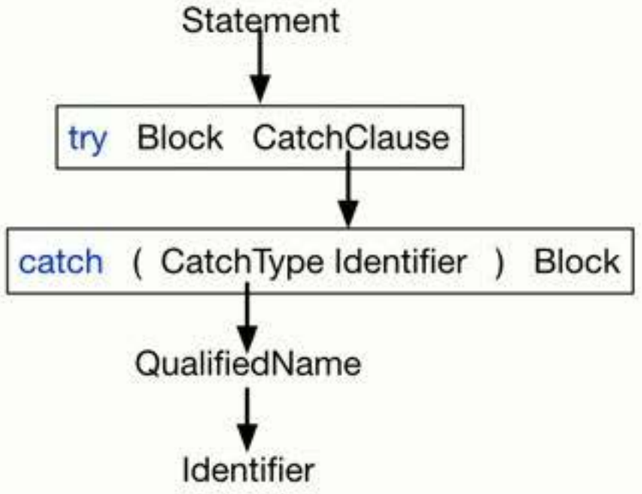
Instantiation of a
new object

Expr → new Identifier (ExprList)



Try-Catch Block

Statement → try Block catch
(Identifier Identifier)
Block

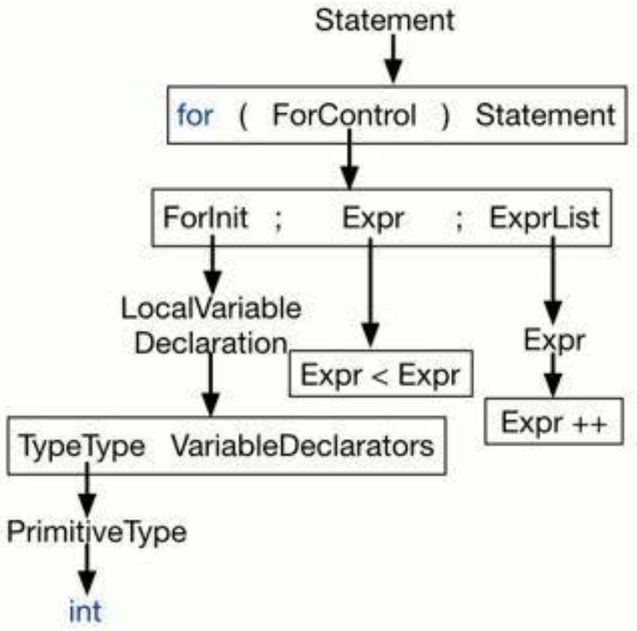


Idioms from CONCODE

Idioms from CONCODE

Integer-based
For loop

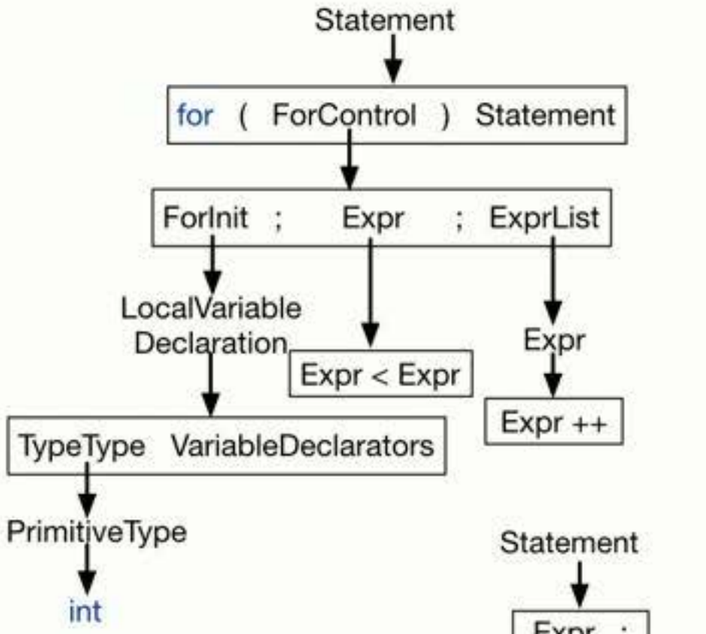
Statement →
for (int VariableDeclarators ;
Expr < Expr ; Expr ++)
Statement



Idioms from CONCODE

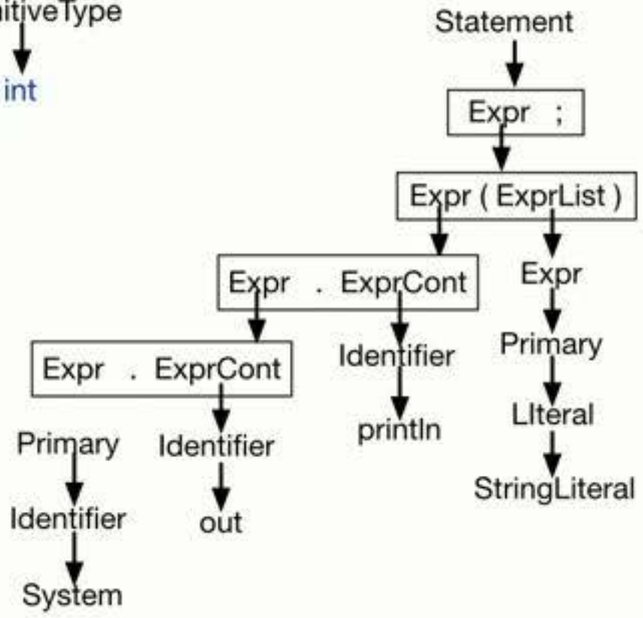
Integer-based **For loop**

Statement →
for (int VariableDeclarators ;
Expr < Expr ; Expr ++)
Statement



Console Output

Statement → System.out.println
(StringLiteral) ;



Idioms from CONCODE

If-then Idiom

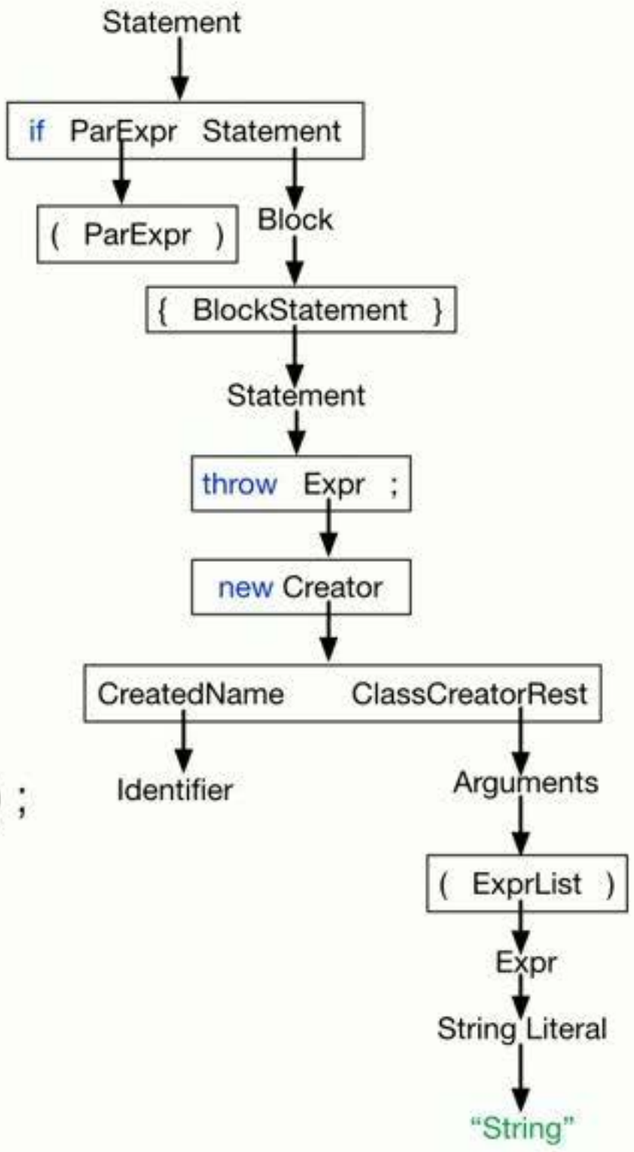
+

Try-Catch Idiom

+

New Object Idiom

```
Statement →
 if ( ParExpr ) {
   throw new Identifier ( "String" ) ;
 }
```



Talk Outline

1. Mapping NL to Source Code in Programmatic Context

   Srini Iyer, Yannis Konstas, Alvin Cheung, Luke Zettlemoyer,
   Mapping Language to Code in Programmatic Context, EMNLP 2018

   Srini Iyer, Alvin Cheung, Luke Zettlemoyer,
   Learning Programmatic Idioms for Scalable Semantic Parsing, EMNLP 2019

2. Learning a Extensible Semantic Parser using User Feedback

   Srini Iyer, Yannis Konstas, Alvin Cheung, Jayant K., Luke Zettlemoyer,
   Learning a Neural Semantic Parser from User Feedback, ACL 2017

3. Topics for future research

   Alane Suhr, **Srini Iyer,** Yoav Artzi [Outstanding Paper]
   Learning to Map Context Dependent Sentences to Executable Formal Queries, NAACL 2018

   Rajas Agashe, **Srini Iyer,** Luke Zettlemoyer
   JuICEe: A Large Scale Distantly Supervised Dataset for Open Domain Context-based Code , EMNLP 2019

Inexpensive and Extensible NL Interfaces

Inexpensive and Extensible NL Interfaces

1. Use a **popular programming language** that a large number of crowd programmers are familiar with

Inexpensive and Extensible NL Interfaces

1. Use a **popular programming language** that a large number of crowd programmers are familiar with

2. Have **accurate models** to map NL → Code

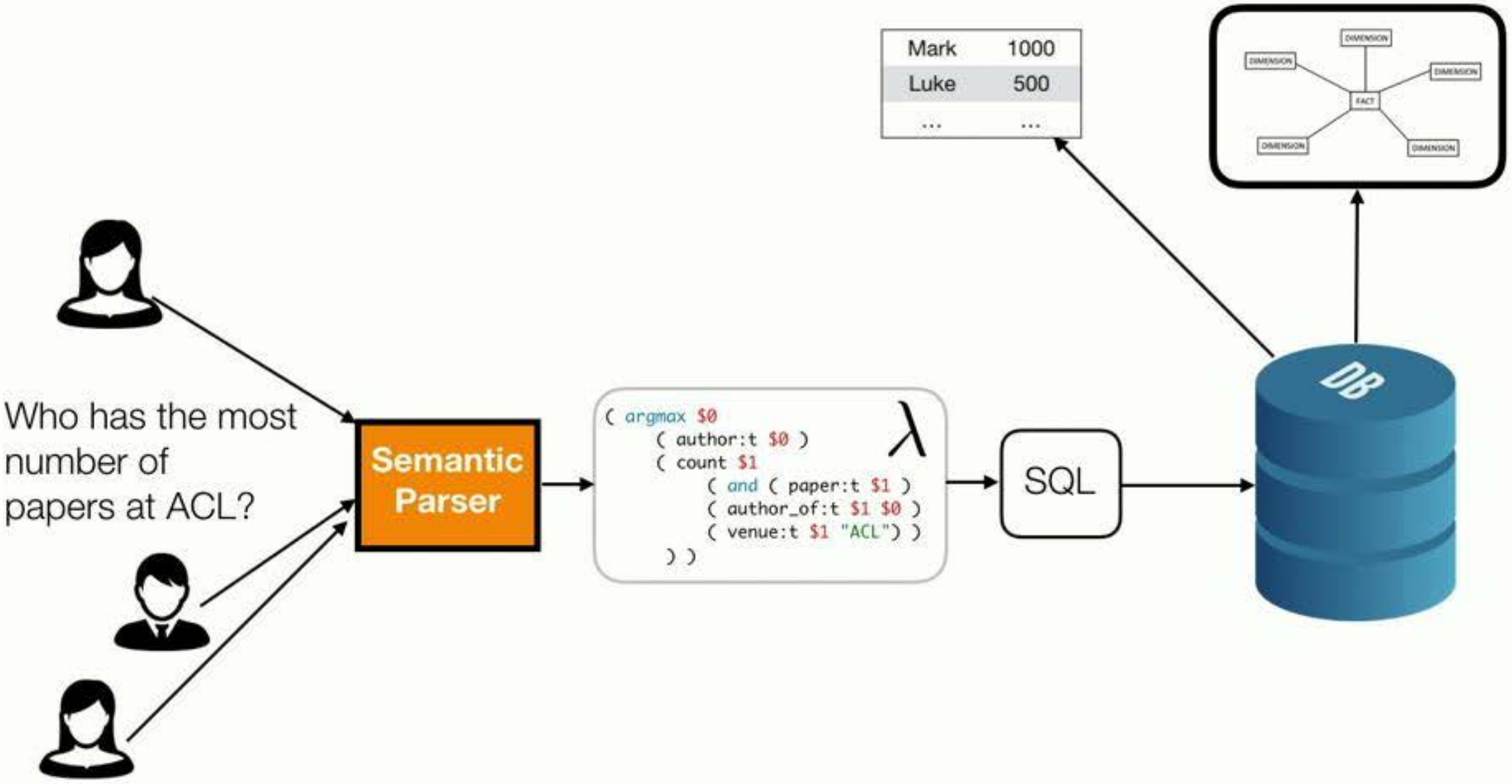
Inexpensive and Extensible NL Interfaces

1. Use a **popular programming language** that a large number of crowd programmers are familiar with

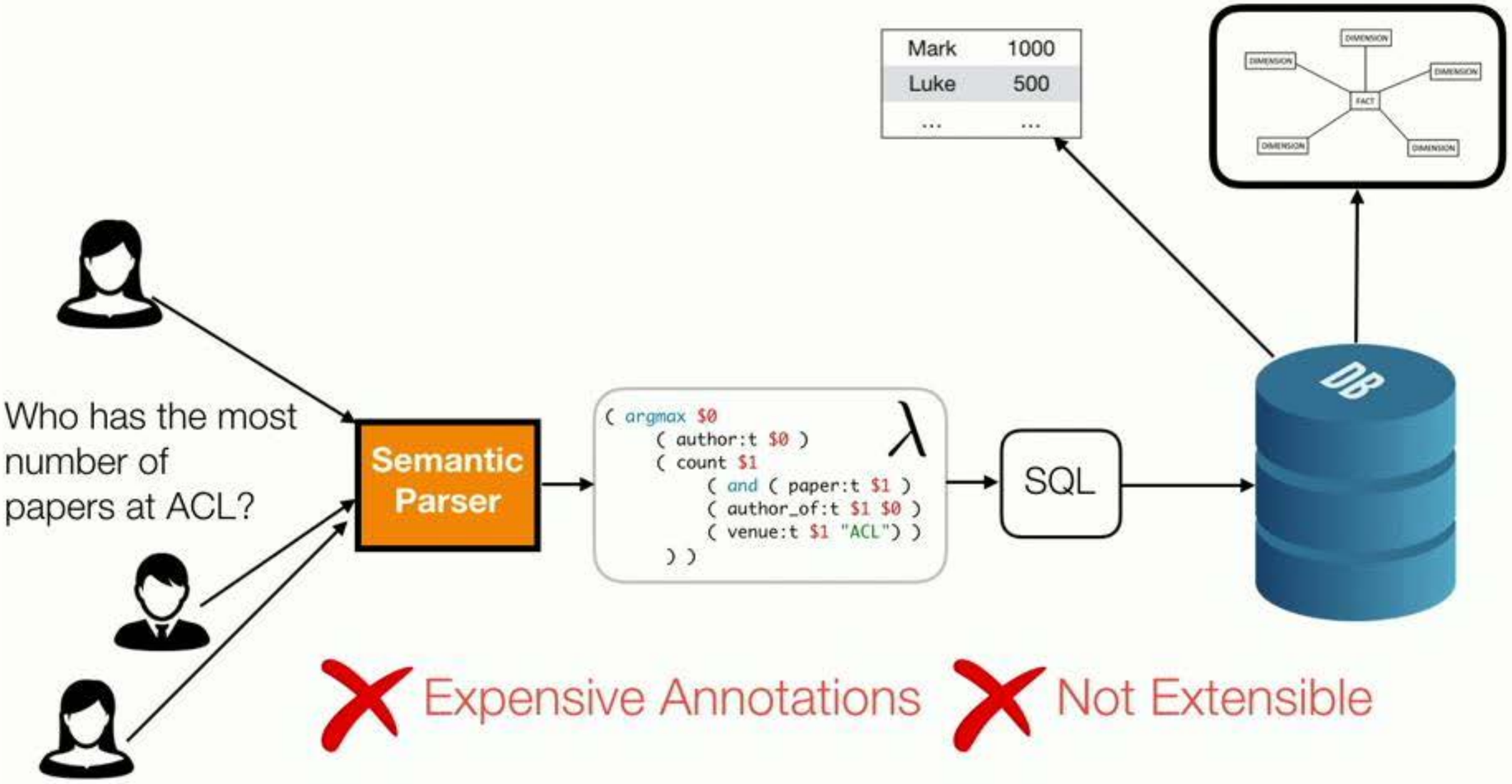
2. Have **accurate models** to map NL → Code

3. Build a mechanism to get **user feedback** and **new annotations** to improve models

Building NL Interfaces for new domains

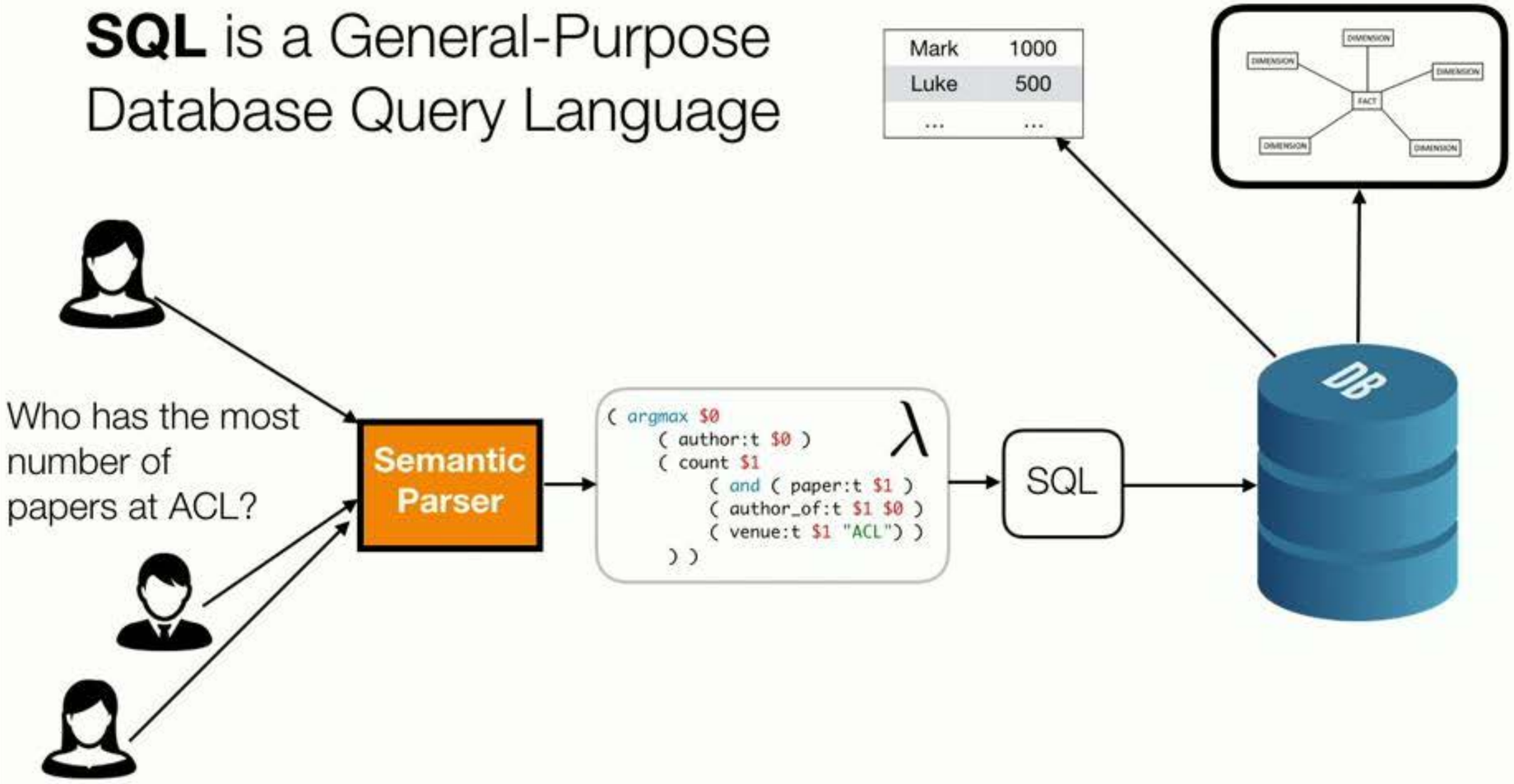


Building NL Interfaces for new domains

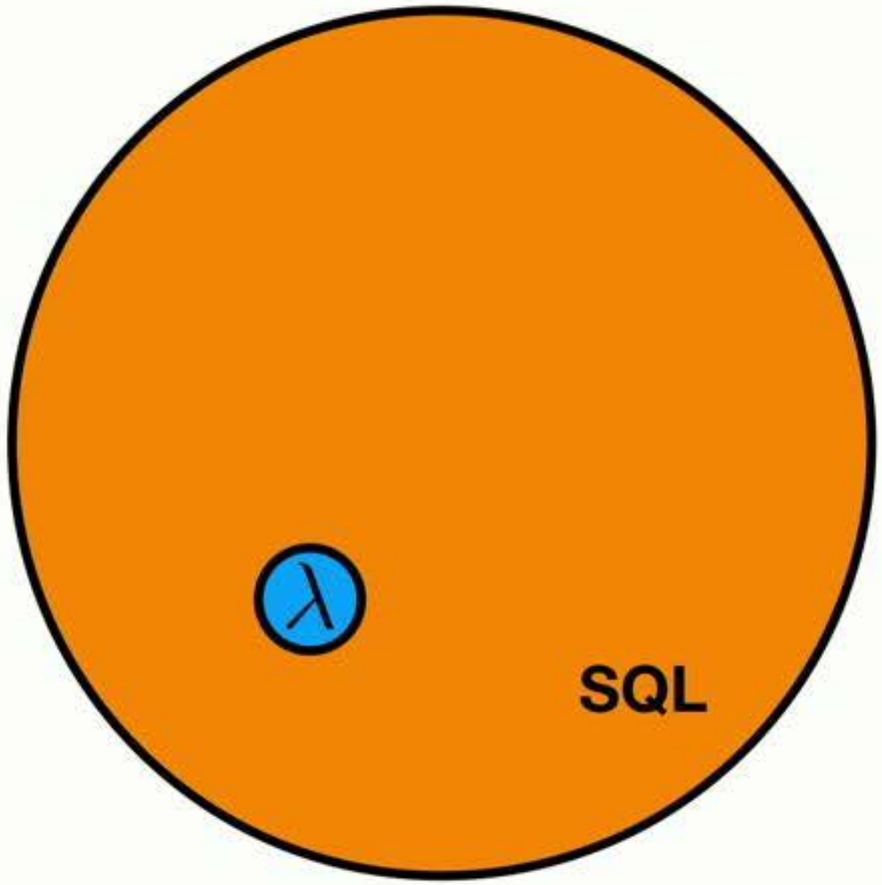


Building NL Interfaces for new domains

SQL is a General-Purpose Database Query Language



Quicker and Cheaper Annotations

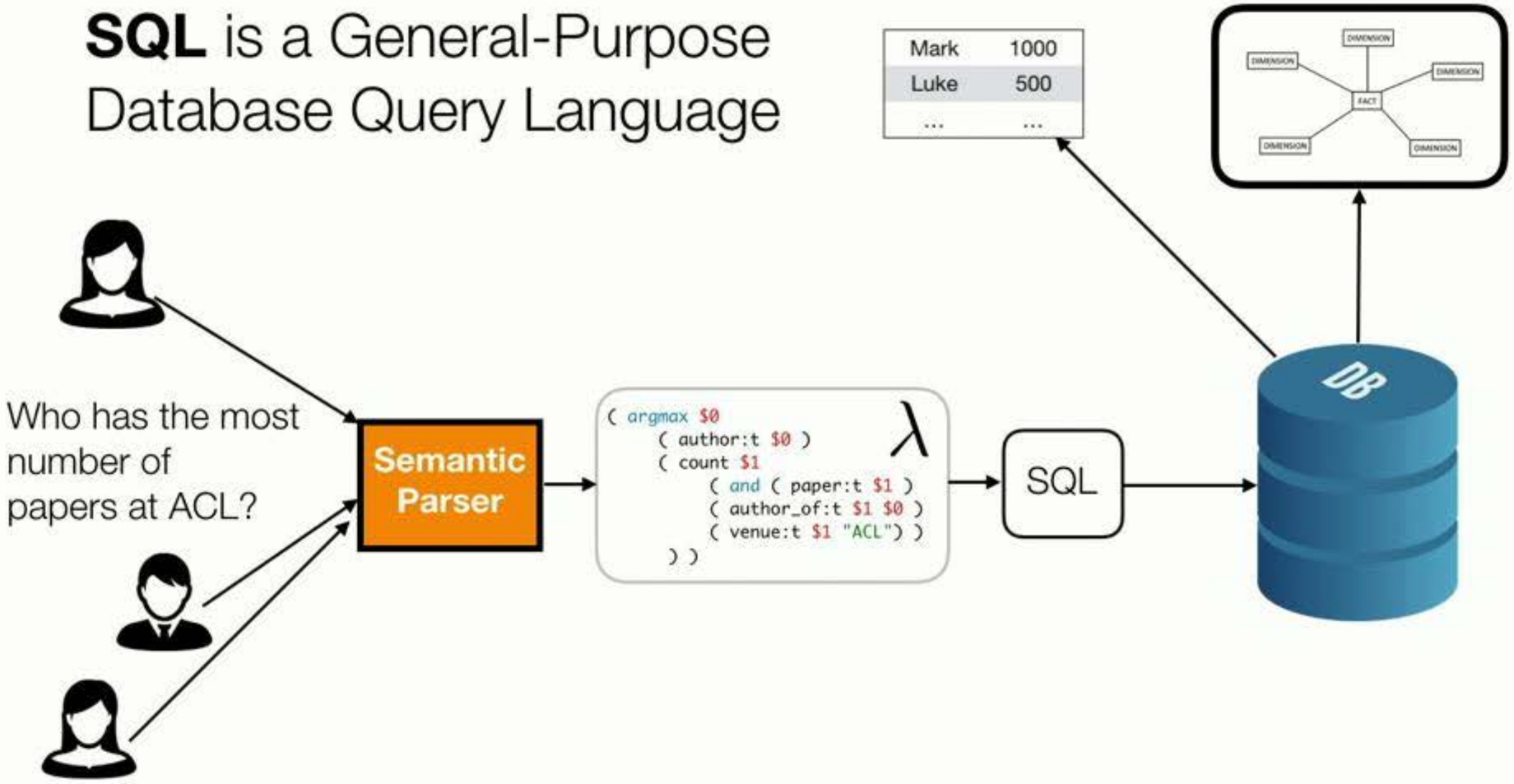


% Crowd Workers
familiar with language

Easy and **Cheap** to find crowd SQL programmers to provide quick annotations

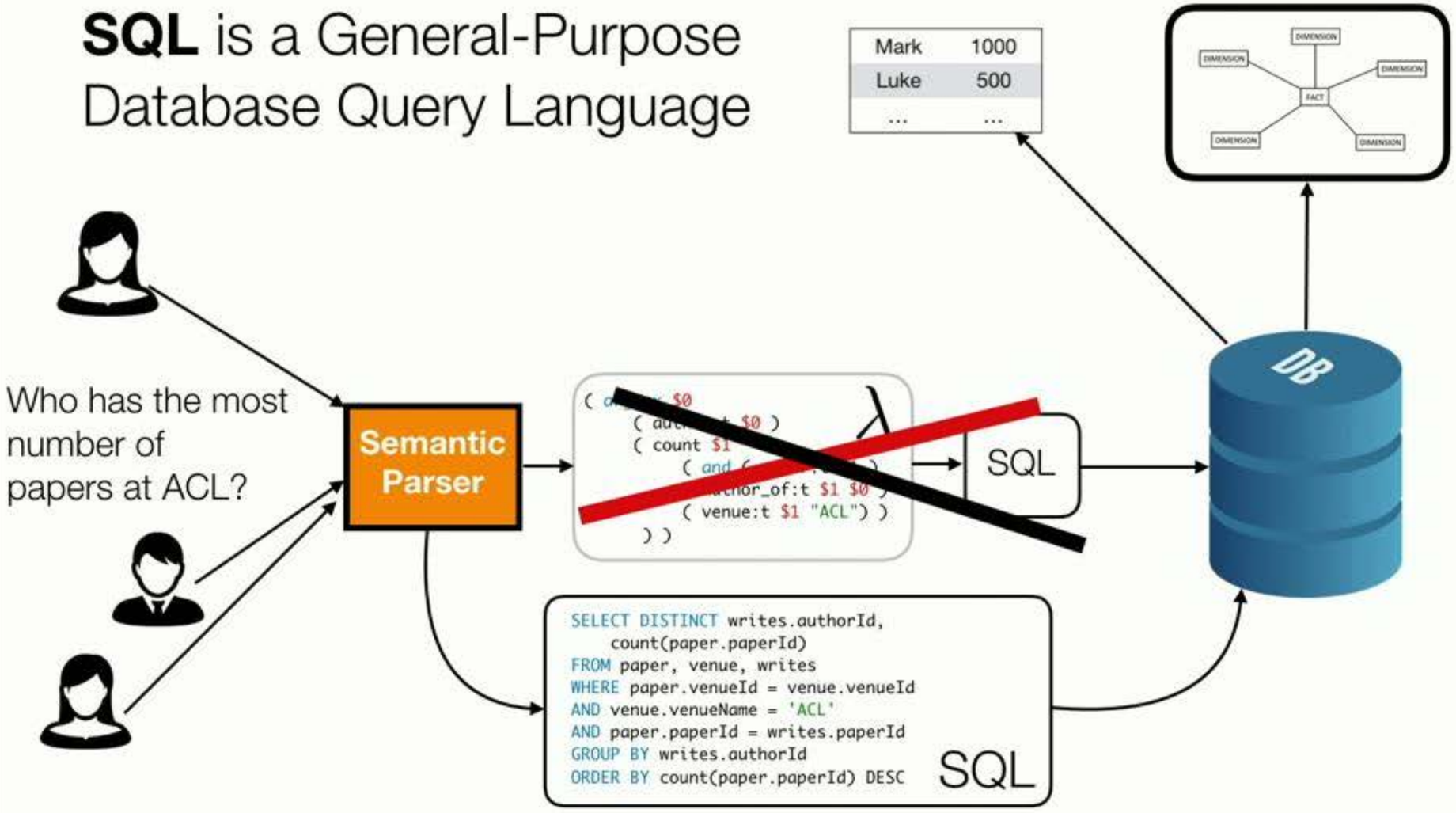
Building NL Interfaces for new domains

SQL is a General-Purpose Database Query Language



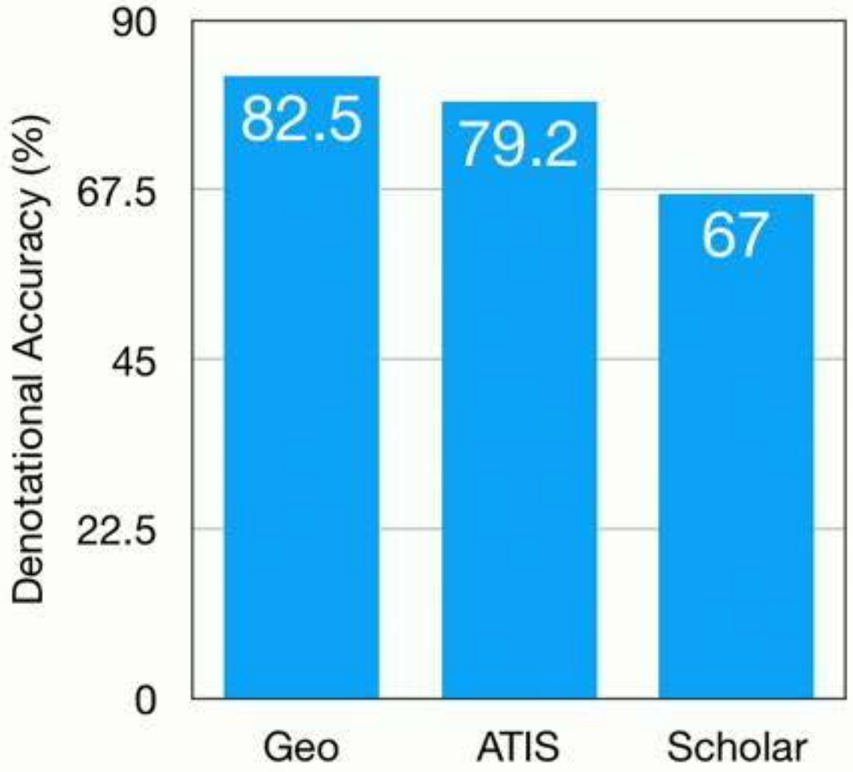
Building NL Interfaces for new domains

SQL is a General-Purpose Database Query Language



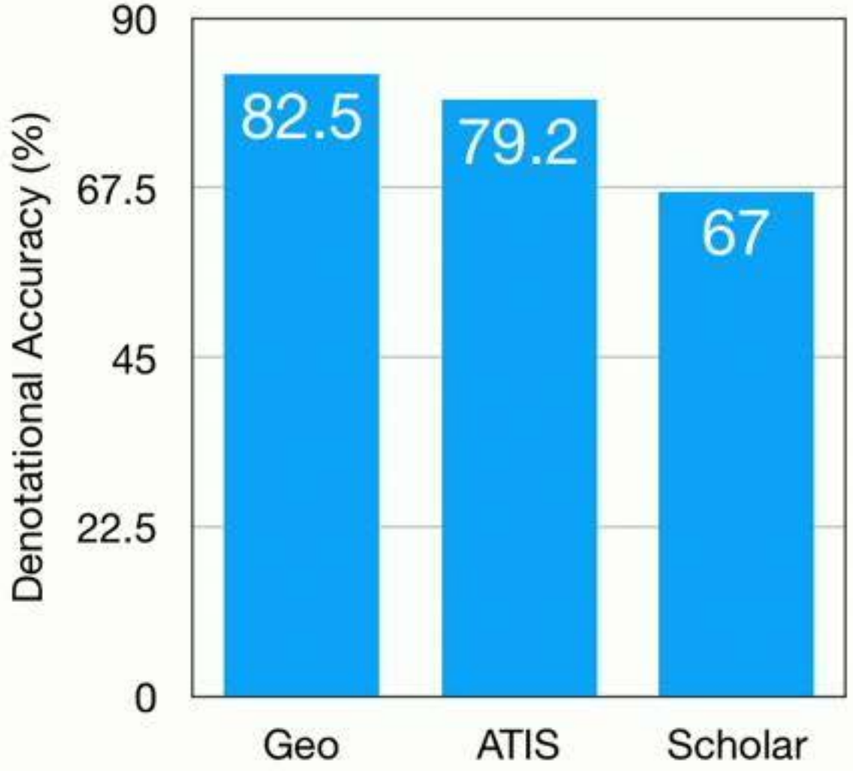
Seq2Seq Models can map NL to SQL

Iyer et al. ACL 2017 show that Seq2Seq models **with paraphrasing** can effectively map NL to SQL



Seq2Seq Models can map NL to SQL

Iyer et al. ACL 2017 show that Seq2Seq models **with paraphrasing** can effectively map NL to SQL

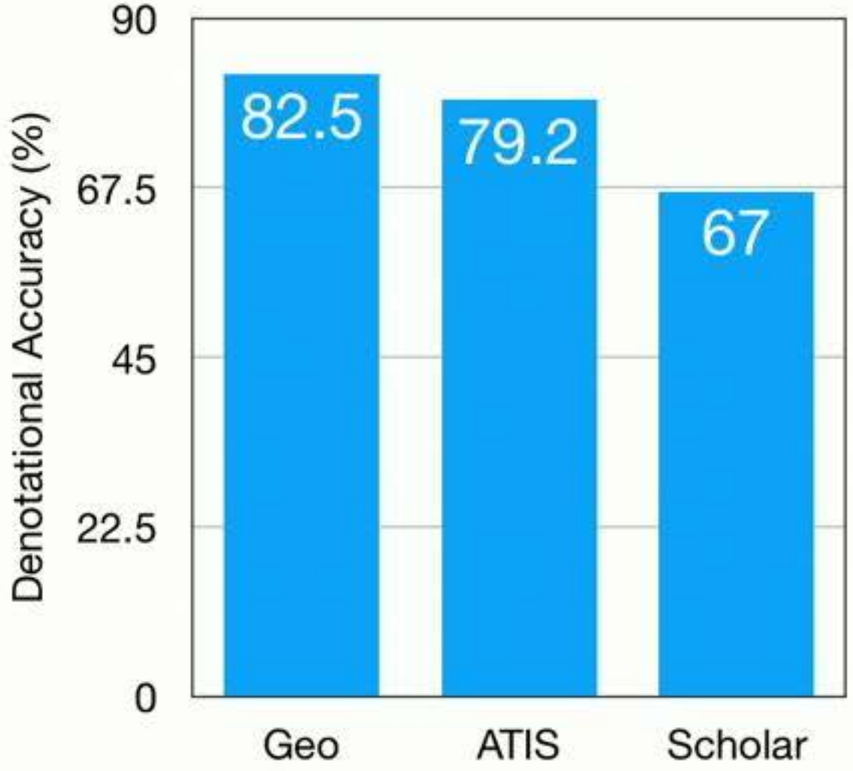


Single-turn Interactions

- WikiSQL (Zhong et al., 2017)
- SQL Evaluation (Finegan-Dollak et al. ACL 2018)
- SPIDER (Yu et al., EMNLP 2018)

Seq2Seq Models can map NL to SQL

Iyer et al. ACL 2017 show that Seq2Seq models **with paraphrasing** can effectively map NL to SQL



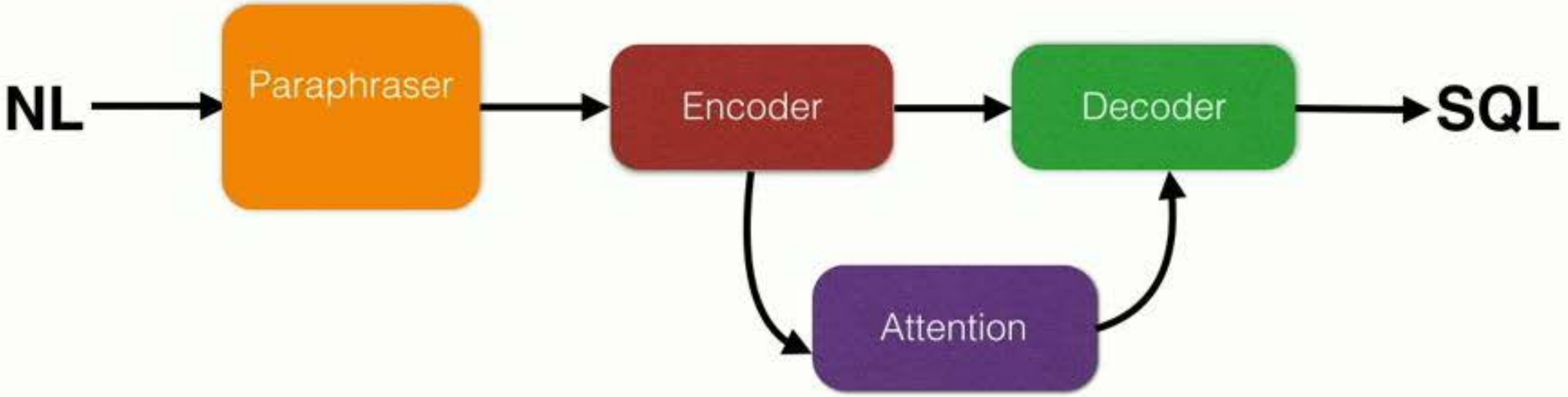
Single-turn Interactions

- WikiSQL (Zhong et al., 2017)
- SQL Evaluation (Finegan-Dollak et al. ACL 2018)
- SPIDER (Yu et al., EMNLP 2018)

Multi-turn Interactions

- Context-Dependent ATIS (**Suhr, Iyer, Artzi** - NAACL 2018) [Outstanding Paper]
- SParC (Yu et al., ACL 2019)

Seq-Seq attention models



Few **SQL query templates**, numerous **NL questions !**

Crowd Programmers to improve models

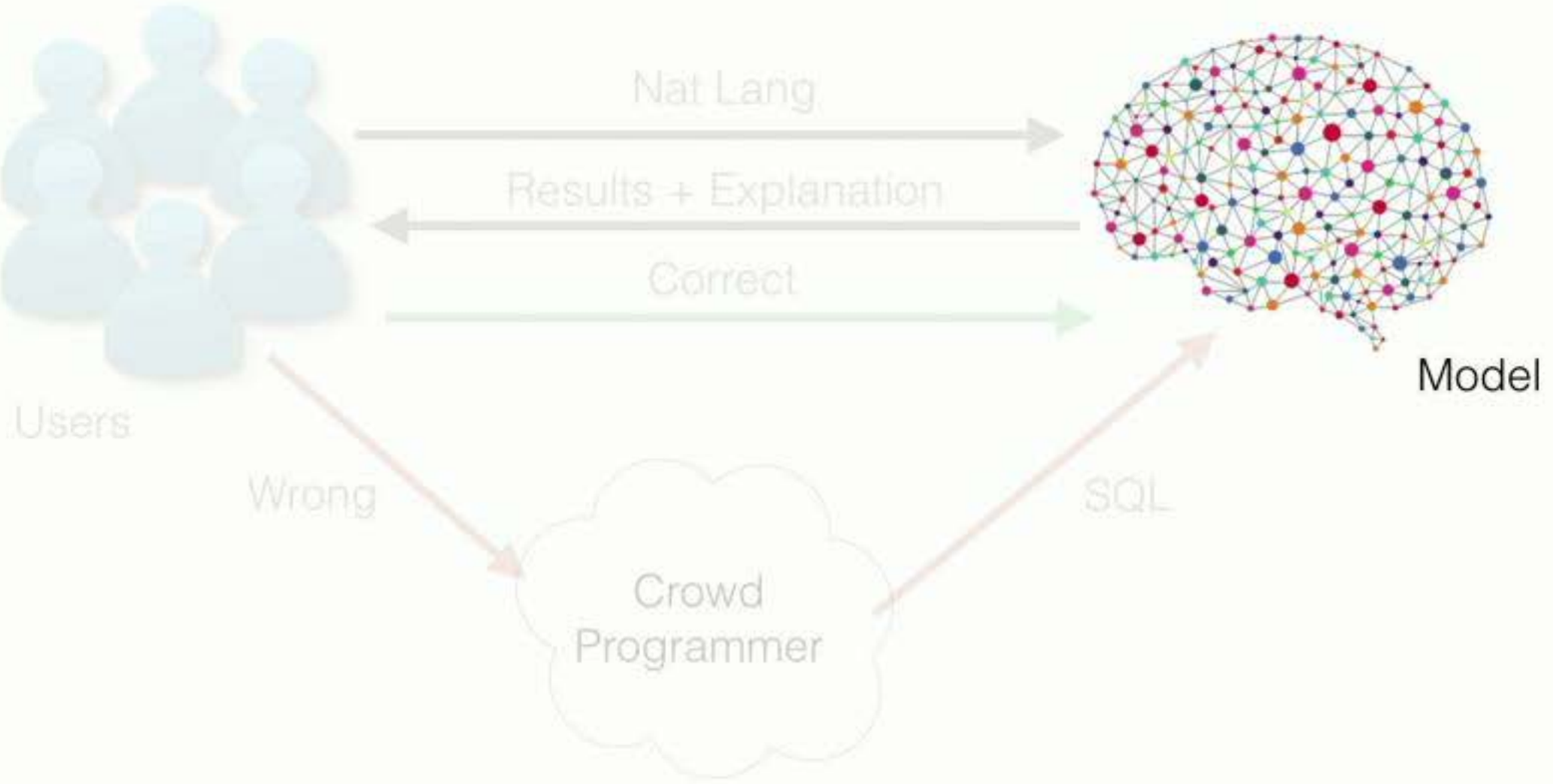
1. Use a **popular programming language** that a large number of crowd programmers are familiar with

2. Have **accurate models** to map NL → Code

3. Build a mechanism to get **user feedback** and **improve models**

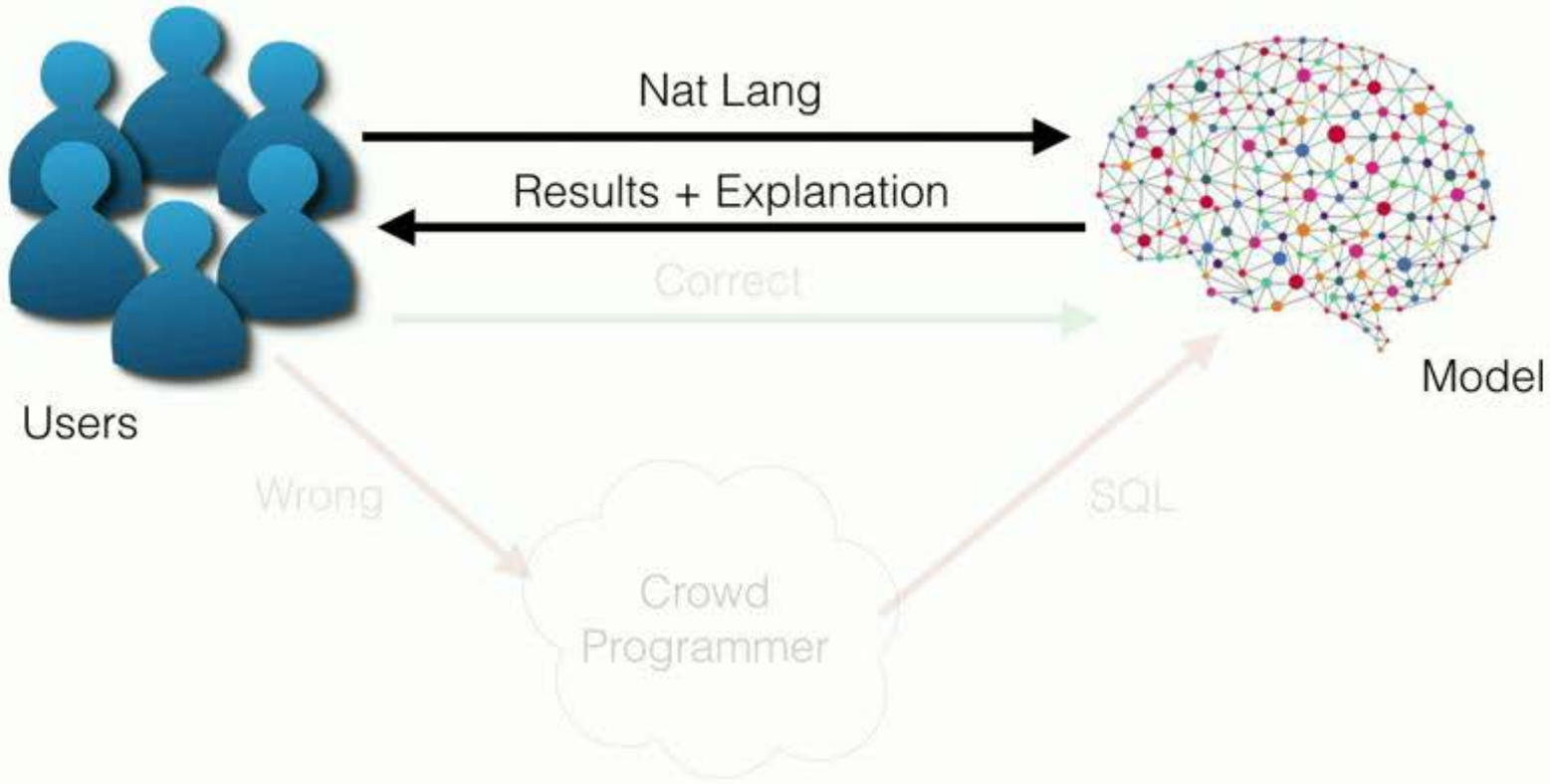
A NL Interface from Scratch

1. Train an initial model to **directly generate SQL** using automatically generated data from **SQL templates**



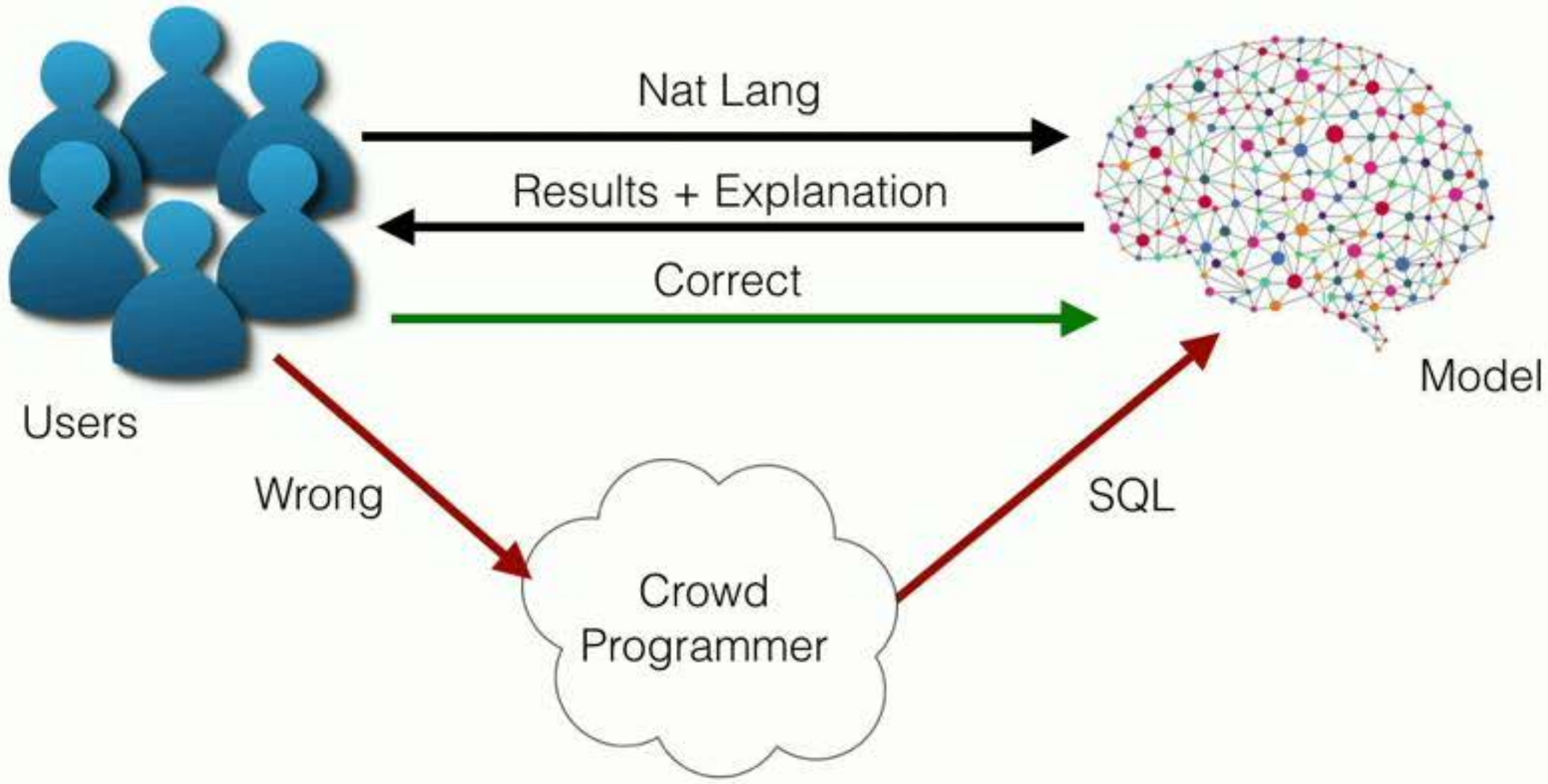
A NL Interface from Scratch

2. Deploy the model in its **target environment.** Users see **results** + **Model explanation**



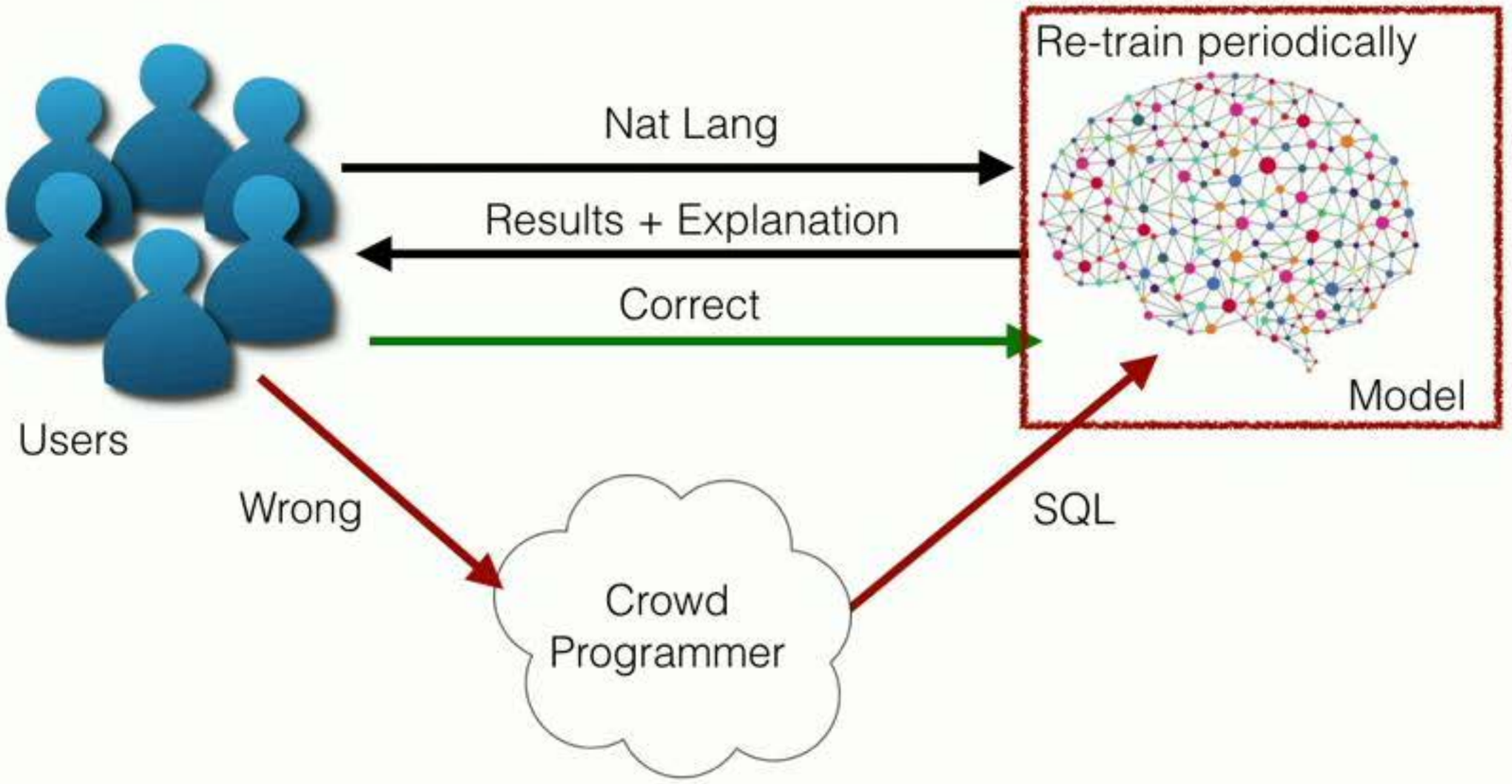
A NL Interface from Scratch

3. Use **crowd programmers** to provide supervision on model's mistakes



A NL Interface from Scratch

4. **Re-train models** periodically



NLIDB for Semantic Scholar

Semantic Scholar is an online
Academic Paper Knowledge Base
https://www.semanticscholar.org/

3-stage experiment

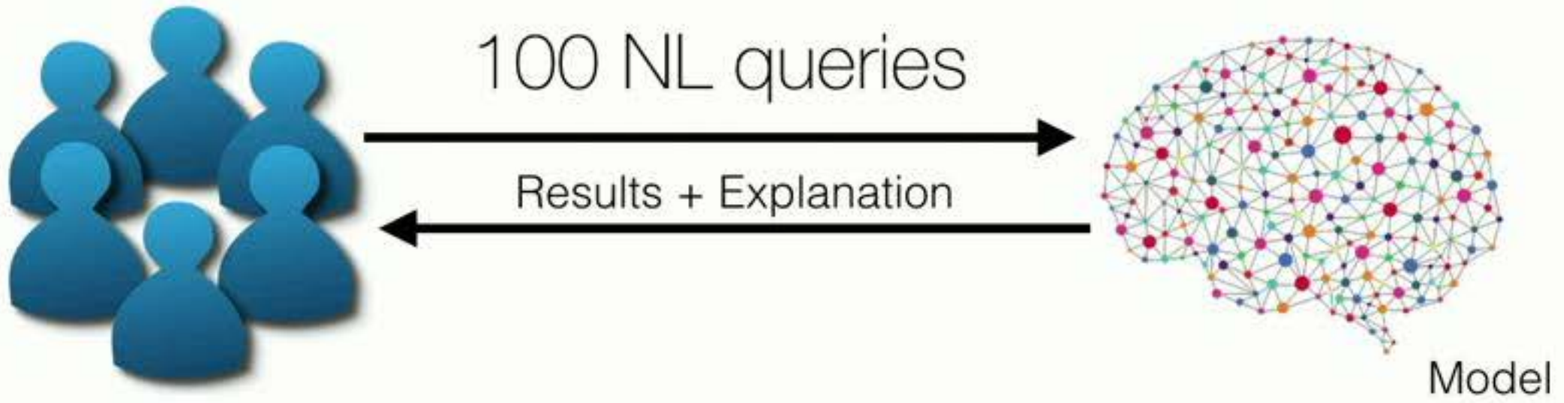
3-stage experiment

10 Users

A **single stage** of the experiment

3-stage experiment

10 Users



A **single stage** of the experiment

Explanations of Results

Type Highlighting

Utterance Paraphrasing

93

Explanations of Results

Type Highlighting

Papers by Michael I. Jordan **(AUTHOR)** in ICRA **(VENUE)** in 2016 **(YEAR)**

Utterance Paraphrasing

93

Explanations of Results

Type Highlighting

Papers by Michael I. Jordan **(AUTHOR)** in ICRA **(VENUE)** in 2016 **(YEAR)**

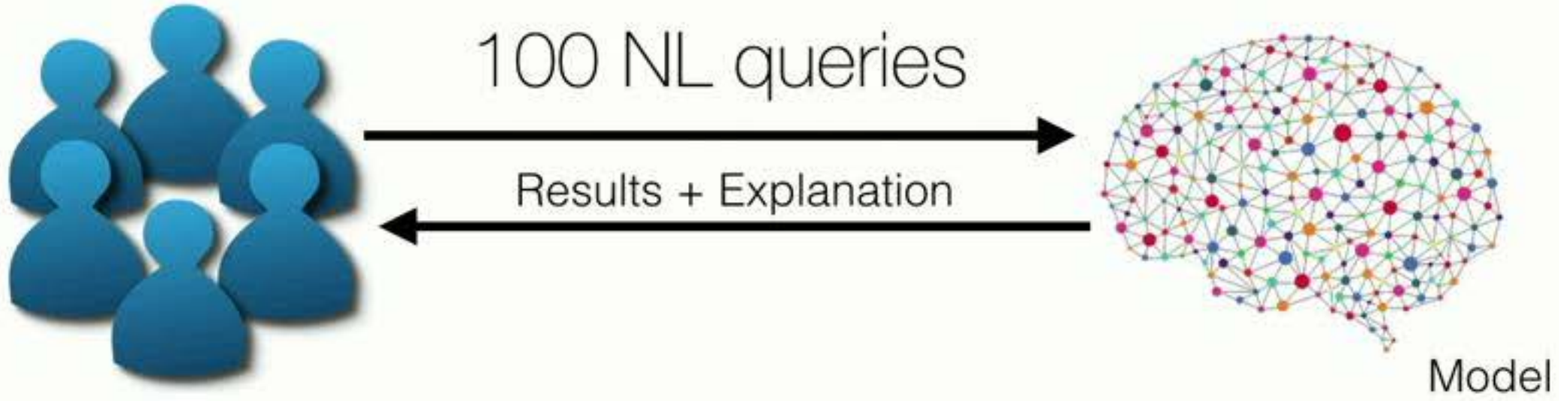
Utterance Paraphrasing

What papers does Michael I. Jordan **(AUTHOR)** have in ICRA **(VENUE)** in 2016 **(YEAR)**

93

3-stage experiment

10 Users

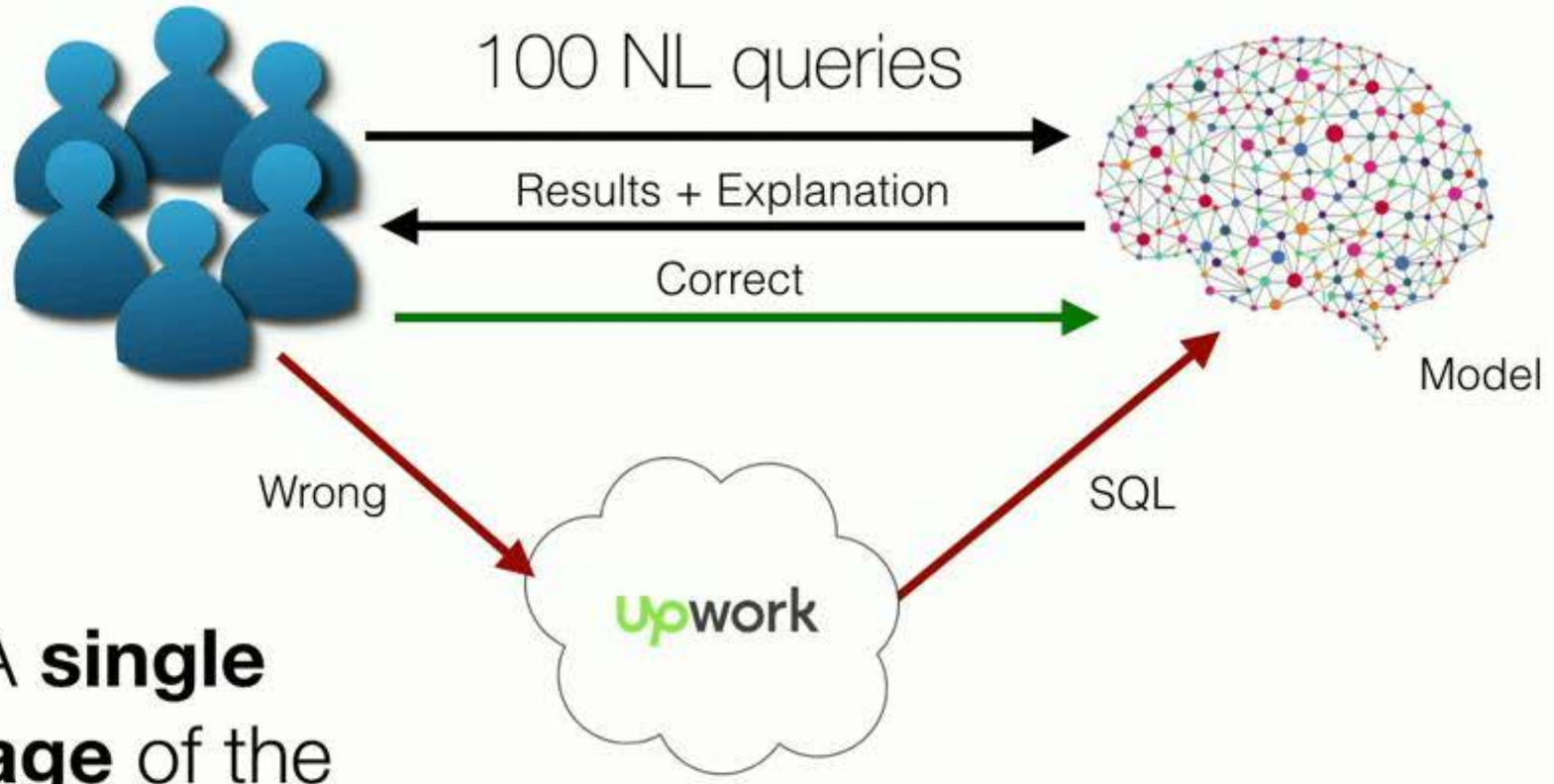


A **single stage** of the experiment

94

3-stage experiment

10 Users

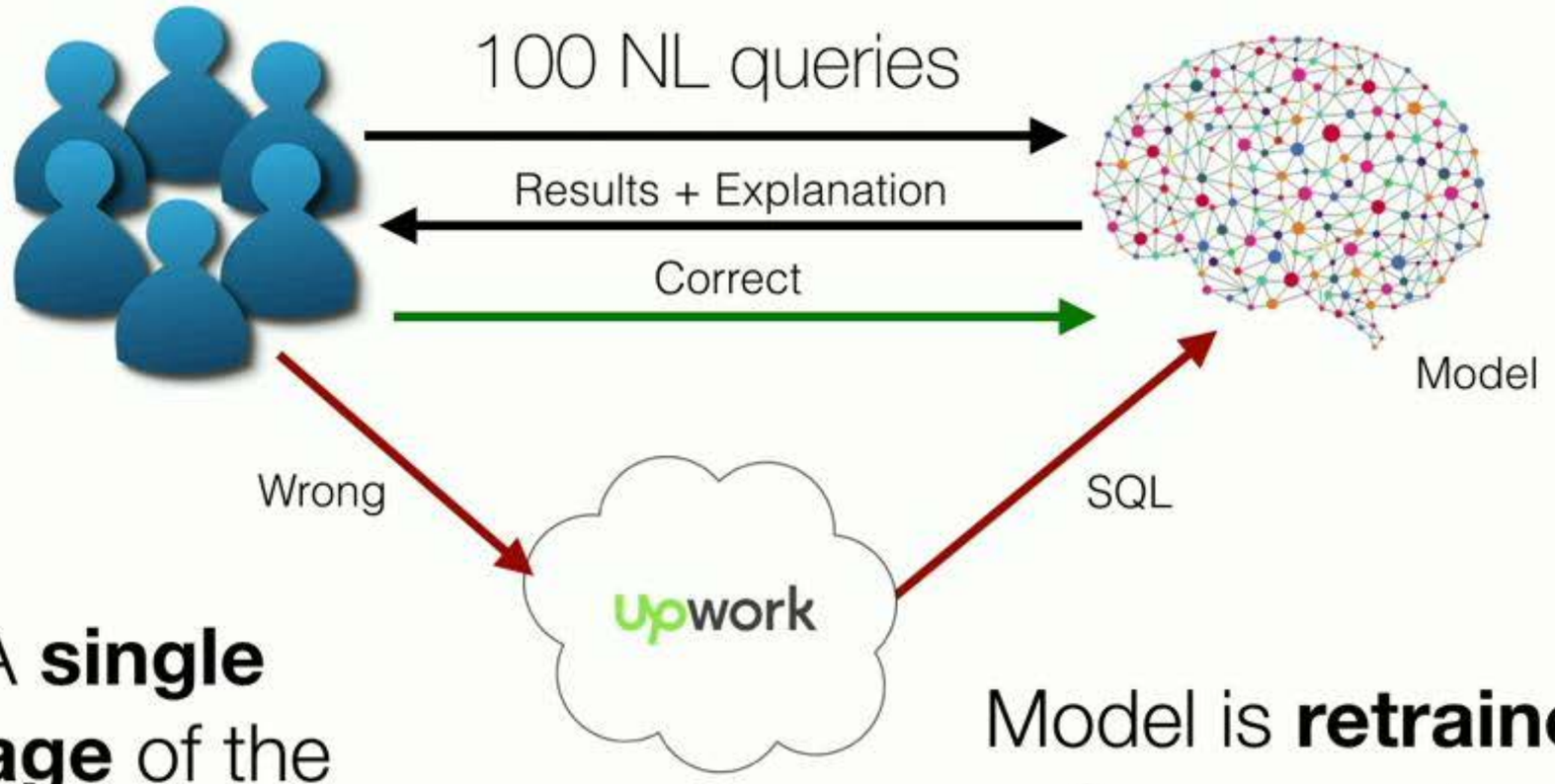


A **single stage** of the experiment

94

3-stage experiment

10 Users



A **single stage** of the experiment

Model is **retrained** after every stage

94

Results

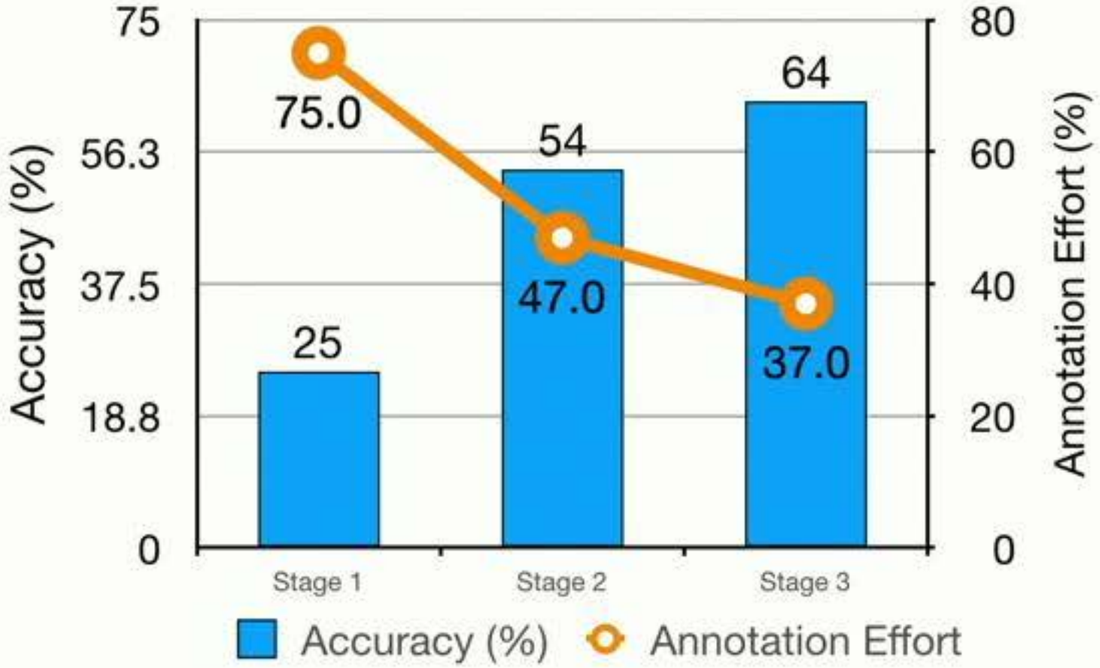
- 3-stage Experiment
- 10 different users/stage
- 100 NL queries/stage

1. **63%** accuracy in 3 stages

2. Annotation effort **reduces** per stage

Model was **retrained** after each stage.

Accuracy (%) - Queries marked by Users as Correct/Incomplete



Talk Outline

1. Mapping NL to Source Code in Programmatic Context

   Srini Iyer, Yannis Konstas, Alvin Cheung, Luke Zettlemoyer,
   Mapping Language to Code in Programmatic Context, EMNLP 2018

   Srini Iyer, Alvin Cheung, Luke Zettlemoyer,
   Learning Programmatic Idioms for Scalable Semantic Parsing, EMNLP 2019

2. Learning a Extensible Semantic Parser using User Feedback

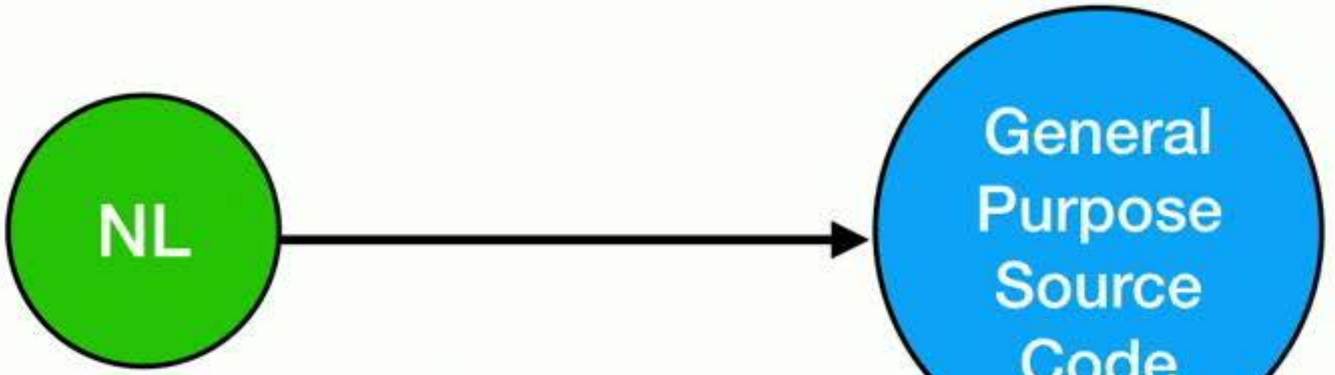
   Srini Iyer, Yannis Konstas, Alvin Cheung, Jayant K., Luke Zettlemoyer,
   Learning a Neural Semantic Parser from User Feedback, ACL 2017

3. Topics for future research

   Alane Suhr, **Srini Iyer,** Yoav Artzi [Outstanding Paper]
   Learning to Map Context Dependent Sentences to Executable Formal Queries, NAACL 2018

   Rajas Agashe, **Srini Iyer,** Luke Zettlemoyer
   JulCEe: A Large Scale Distantly Supervised Dataset for Open Domain Context-based Code , EMNLP 2019

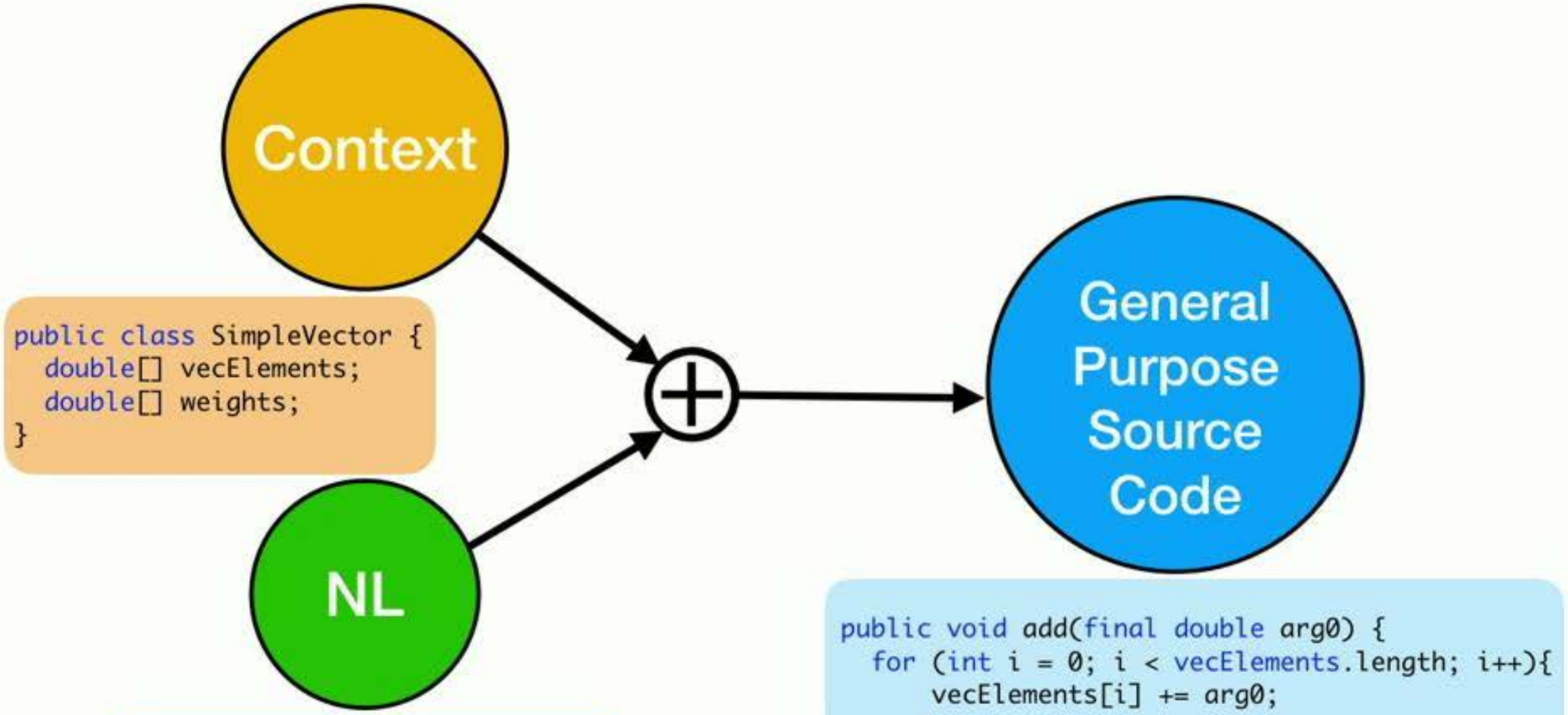
NL→Code



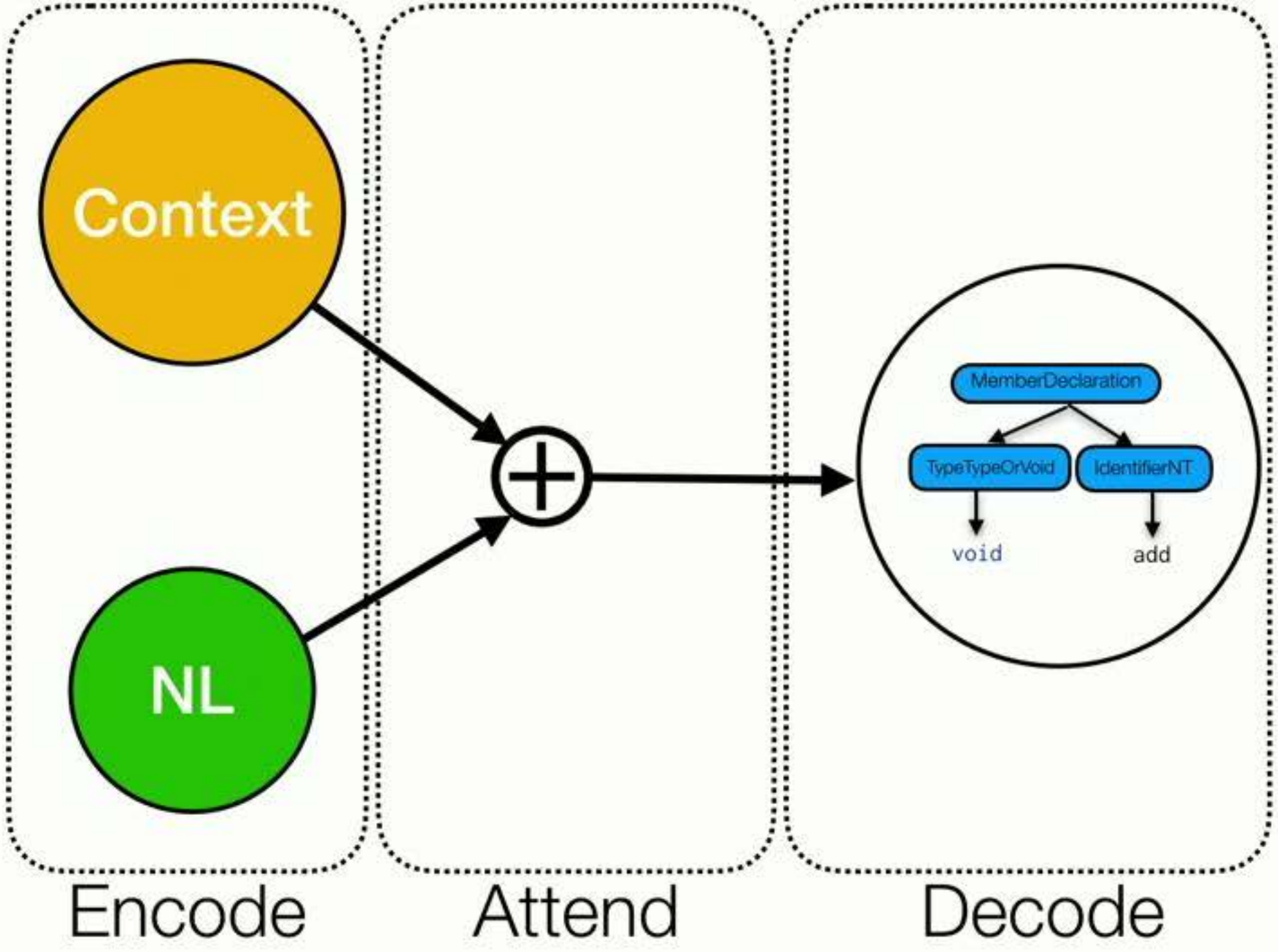
NL Query: Adds a scalar to this vector in place.

```
public void add(final double arg0) {
  for (int i = 0; i < vecElements.length; i++){
      vecElements[i] += arg0;
    }
  }
```

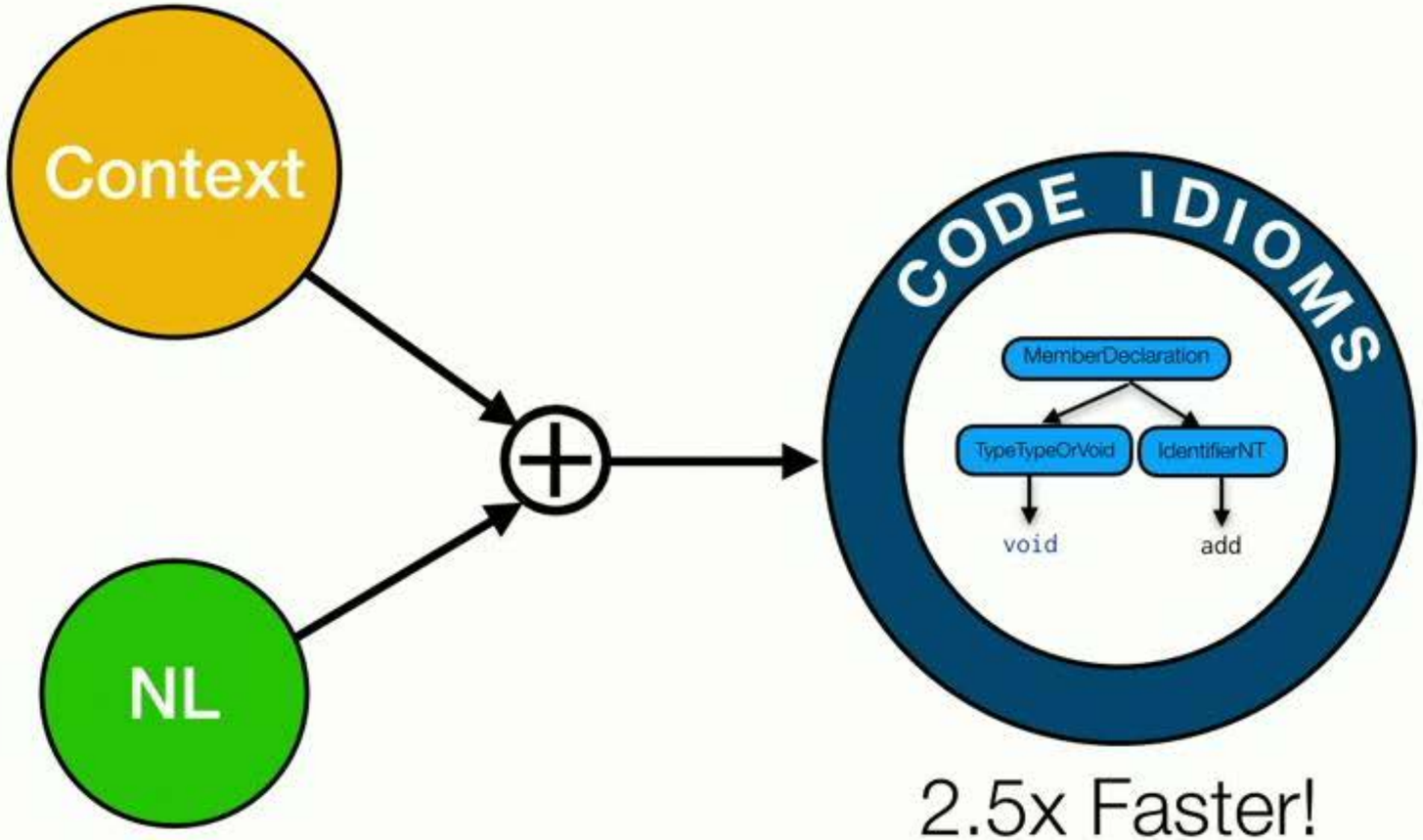
Context is vital!



Specialized Enc-Attend-Dec Models

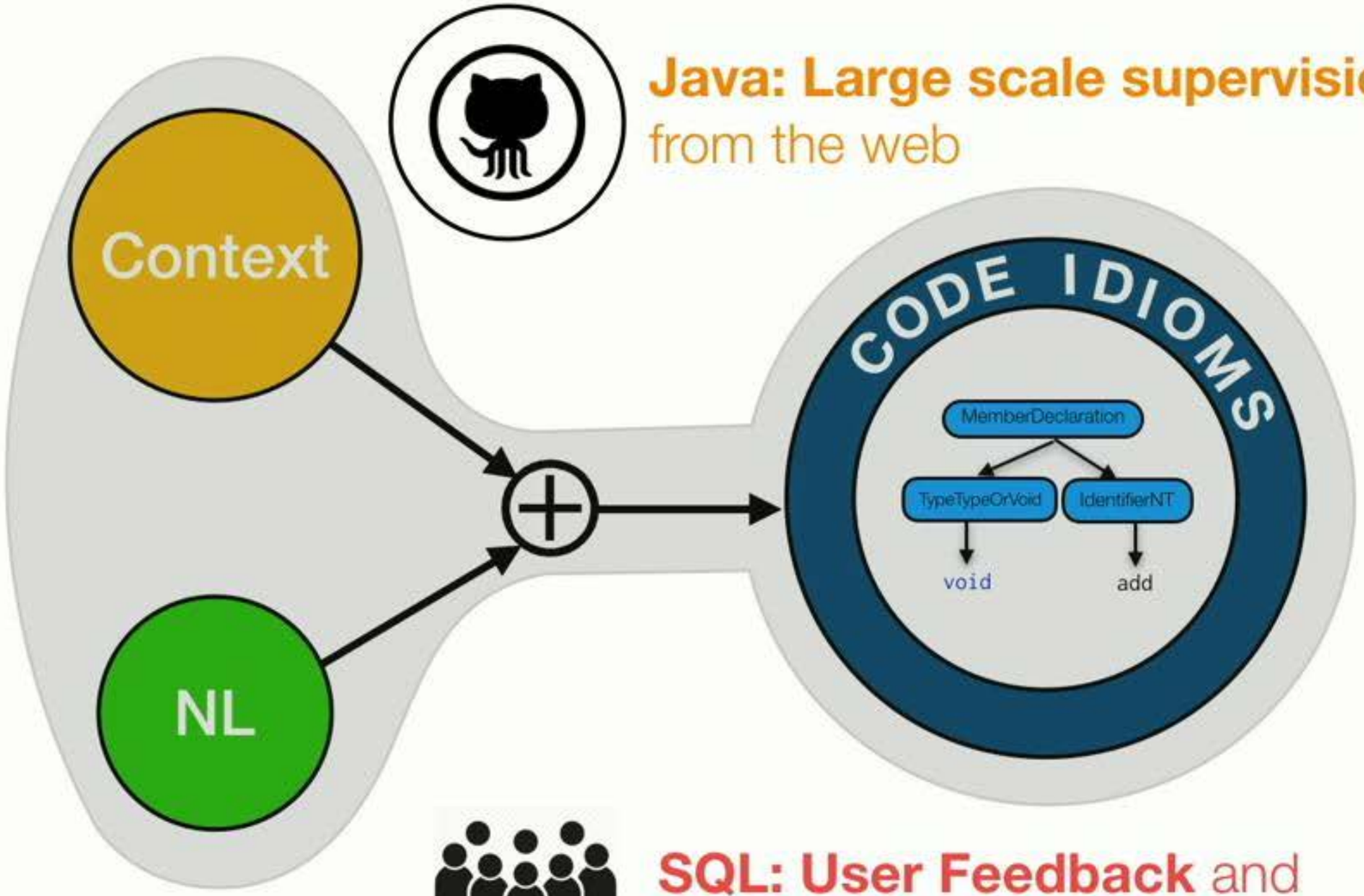


Code Idioms



2.5x Faster!

Corpora for training models



Java: Large scale supervision from the web

SQL: User Feedback and **Crowdprogramming**

Learning NL→Code Models at Scale

Dataset	# Examples
Geoquery	880
ATIS	5,500
SPIDER	10,181
CONCODE	500,000
CoNaLa	600,000
Github	**> 10 Million**

Lots of unused data available on the web

Learning NL→Code Models at Scale

Dataset	# Examples
Geoquery	880
ATIS	5,500
SPIDER	10,181
CONCODE	500,000
CoNaLa	600,000
Github	**> 10 Million**

Lots of unused data available on the web

1. **Faster, distributed** models for large datasets
   - Attention-based models that can decode syntactically correct Code?

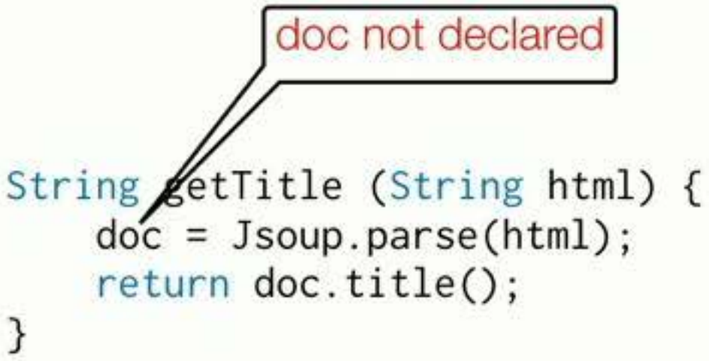
Learning NL→Code Models at Scale

Dataset	# Examples
Geoquery	880
ATIS	5,500
SPIDER	10,181
CONCODE	500,000
CoNaLa	600,000
Github	**> 10 Million**

Lots of unused data available on the web

1. **Faster, distributed** models for large datasets
   - Attention-based models that can decode syntactically correct Code?
2. **Pretrained Representations** for Idioms and APIs?

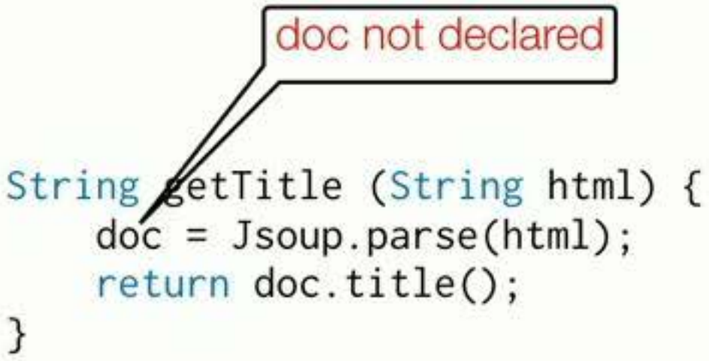
Learning NL→Code Models at Scale



3. **Semantic Constraints** while decoding (Lin et al., 2019)

- Variables/Methods should be defined before being used
- Type Checking

Learning NL→Code Models at Scale



3. **Semantic Constraints** while decoding (Lin et al., 2019)

   - Variables/Methods should be defined before being used
   - Type Checking

4. **Introducing Invariants** while training
   - Order of predicates doesn't matter
   - Order of statements etc.

Explaining Predictions to Users

1. Improve **trust.** The User and the Model are a **team.**

Explaining Predictions to Users

Show me all papers from **PLDI 2014**

```
SELECT DISTINCT paper.paperId
FROM paper , venue
WHERE paper.venueId = venue.venueId
AND venue.venueName = "pldi"
```

Get me all papers from **PLDI?**

1. Improve **trust.** The User and the Model are a **team.**

2. Particularly important for **non-expert users.**

Explaining Predictions to Users

Show me all papers from **PLDI 2014**

```
SELECT DISTINCT paper.paperId
FROM paper , venue
WHERE paper.venueId = venue.venueId
AND venue.venueName = "pldi"
```

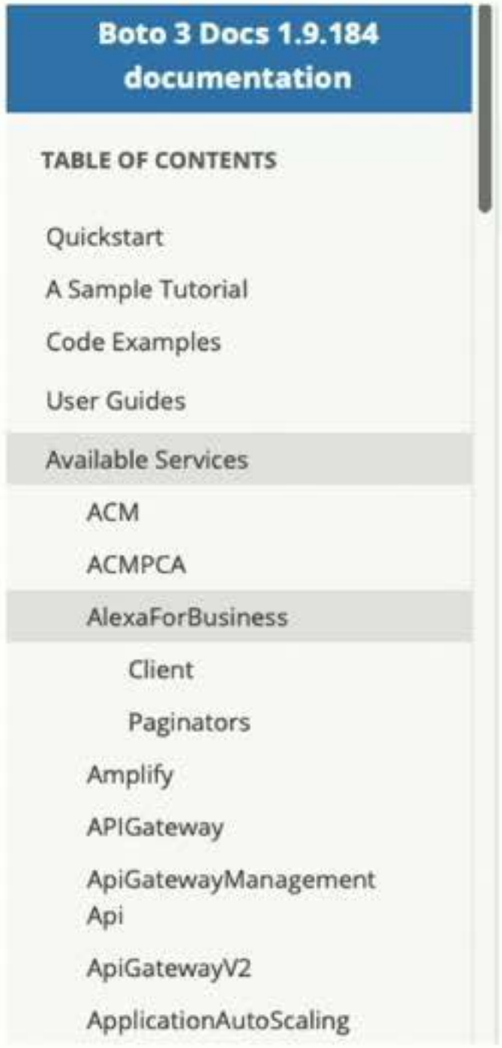
Get me all papers from **PLDI?**

1. Improve **trust.** The User and the Model are a **team.**
2. Particularly important for **non-expert users.**
3. **Context Independent Explanations**

   Srini Iyer, Yannis Konstas, Alvin Cheung, Luke Zettlemoyer
   Summarizing Source Code using a Neural Attention Model
   ACL 2016

4. **Context** and **Interaction Dependent** Explanations

Learning from Background Knowledge



1. Not possible to get training examples for **every API call**
2. Build models that **mimic Stackoverflow responders**
3. Learn API usage from **web documentation**

Learning from Background Knowledge

- `approve_skill()`
- `associate_contact_with_address_book()`
- `associate_device_with_network_profile()`
- `associate_device_with_room()`
- `associate_skill_group_with_room()`
- `associate_skill_with_skill_group()`
- `associate_skill_with_users()`
- `can_paginate()`
- `create_address_book()`
- `create_business_report_schedule()`
- `create_conference_provider()`
- `create_contact()`
- `create_gateway_group()`
- `create_network_profile()`
- `create_profile()`
- `create_room()`
- `create_skill_group()`
- `create_user()`
- `delete_address_book()`
- `delete_business_report_schedule()`
- `delete_conference_provider()`
- `delete_contact()`
- `delete_device()`

1. Not possible to get training examples for **every API call**
2. Build models that **mimic Stackoverflow responders**
3. Learn API usage from **web documentation**

Learning from Background Knowledge

delete_device_usage_data(***kwargs*)

When this action is called for a specified shared device, it allows authorized users to delete the device's entire previous history of voice input data and associated response data. This action can be called once every 24 hours for a specific shared device.

When this action is called for a specified shared device, it allows authorized users to delete the device's entire previous history of voice input data. This action can be called once every 24 hours for a specific shared device.

See also: AWS API Documentation

Request Syntax

```
response = client.delete_device_usage_data(
    DeviceArn='string',
    DeviceUsageType='VOICE'
)
```

Parameters

- **DeviceArn** (*string*) -- **[REQUIRED]**

  The ARN of the device.

- **DeviceUsageType** (*string*) -- **[REQUIRED]**

  The type of usage data to delete.

Return type

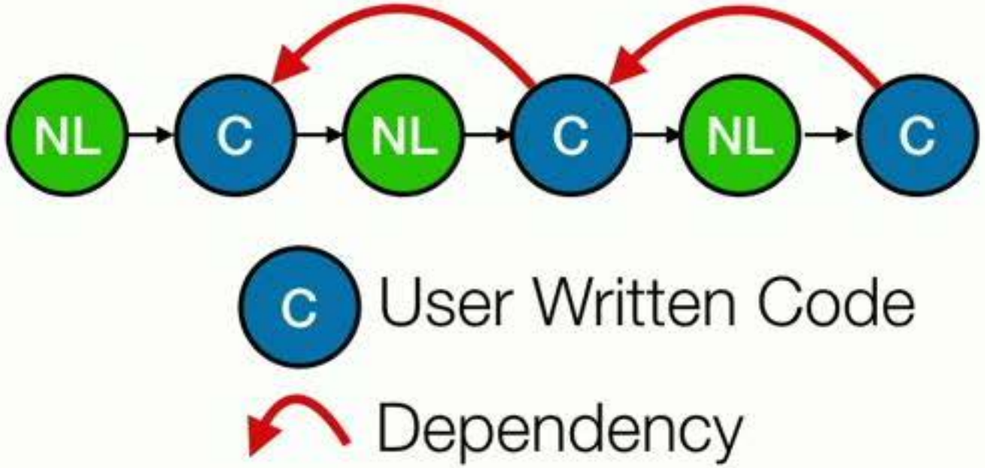
dict

Main Challenges

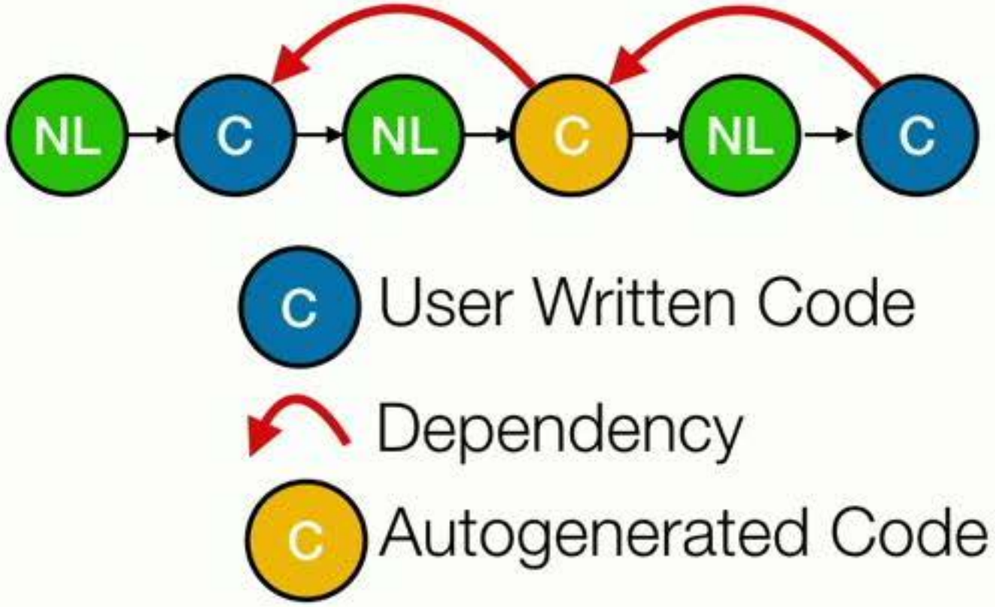
1. **Mismatch** between documentation style and NL Query style

2. Learning to **putting multiple APIs together** to achieve a higher level goal

Interactive Programming Agents



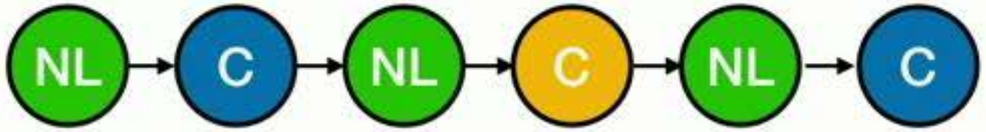
Interactive Programming Agents



User and System are a **team.**

Interactive Computing Agents

Training a Decision Tree



Load features and labels in a DataFrame

```
import pandas as pd
X = pd.read_json('features.json')
y = pd.read_json('labels.json')
```

C User Written Code

C Autogenerated Code

Split the data into train and test

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

Create and train the model

```
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier()
dtree.fit(X_train, y_train)
```

Rajas Agashe, **Srini Iyer,** Luke Zettlemoyer
JuICE: A Large Scale Distantly Supervised Dataset for Open Domain Context-based Code ,
EMNLP 2019

My Amazing Collaborators!

Luke Zettlemoyer
Associate Prof., UW
Research Manager, FAIR

Alvin Cheung
Asst. Prof.,
UC Berkeley

Yannis Konstas
Asst Prof.,
Heriot Watt University

Jayant Krishnamurthy
Researcher,
Semantic Machines

UWNLP

AI2 ALLEN INSTITUTE *for* ARTIFICIAL INTELLIGENCE

facebook

```
// I know this is bad practice. I know, please don't hate me.
// It is 2am and I just need to patch this.
```

```
int random() {
     return 5; // Chosen by dice roll, guaranteed to be random
}
```

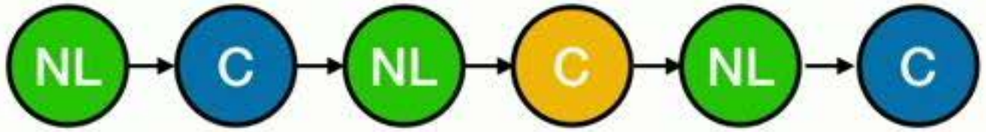
Thank you! Questions?

```
// I can't divide with zero, so I have to divide with something
very similar
result = number / 0.000000000000001;
```

```
Thread.sleep(4000); //make it look like work is being done
```

Interactive Computing Agents

Training a Decision Tree



Load features and labels in a DataFrame

```
import pandas as pd
X = pd.read_json('features.json')
y = pd.read_json('labels.json')
```

C User Written Code

C Autogenerated Code

Split the data into train and test

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

Create and train the model

```
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier()
dtree.fit(X_train, y_train)
```

Rajas Agashe, **Srini Iyer,** Luke Zettlemoyer
JuICE: A Large Scale Distantly Supervised Dataset for Open Domain Context-based Code ,
EMNLP 2019