CrystalBLEU:

Precisely and Efficiently Measuring the Similarity of Code

Aryaz Eghbali, Michael Pradel
Software Lab, University of Stuttgart

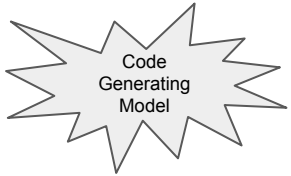
European Research Council
Established by the European Commission

ASE – Oct. 11, 2022
ACM SIGSOFT Distinguished Paper Award

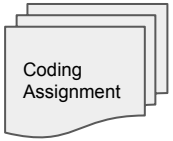
SOLA SoftwareLab
https://software-lab.org/

Motivation

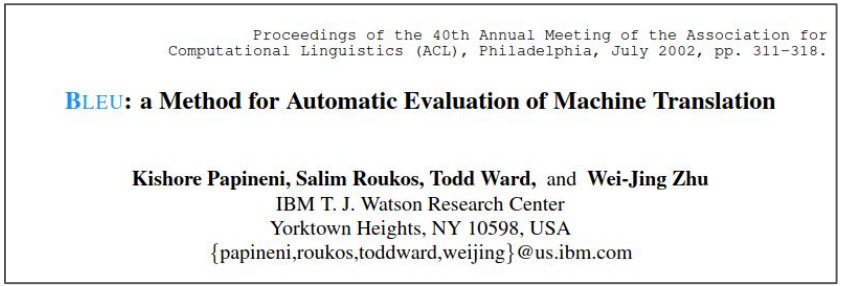
Me



Michael



ctrl+c ctrl+v



Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 311-318.

BLEU: a Method for Automatic Evaluation of Machine Translation

Kishore Papineni, Salim Roukos, Todd Ward, and **Wei-Jing Zhu**
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{papineni,roukos,toddward,weijing}@us.ibm.com

BLEU

Designed for Natural Language (spec. machine translation)

n-grams → matching n-grams → clipped count

$$p_n = \frac{\sum_{C \in Candidates} \sum_{ngram \in C} Count_{clip}(ngram)}{\sum_{C' \in Candidates} \sum_{ngram' \in C'} Count(ngram')}$$

maximum length of n-grams

$$BLEU = exp\left(min(1 - \frac{r}{c}, 0) + \sum_{i=1}^{maxN} w_i \log p_i\right)$$

brevity penalty

weighted average of modified precisions

```
// Reference:
import java.util.*;
public class Main {
  public static void main(String[] args ) {
    Scanner in = new Scanner(System.in);
    int t = in.nextInt();
    in.nextLine();
    while ( t-- > 0 ) {
      System. out.println( new StringBuffer(in.nextLine()).reverse
      ());
    }
  }
}
// Hypothesis 1: equivalent to the reference
import java.util.Scanner;
public class Main {
  public static void main(String argv[] ) {
    int num_of_tests = 0;
    Scanner in = new Scanner(System.in);
    num_of_tests= Integer.parseInt(in.nextLine());
    for(int i=0; i<num_of_tests; i++ ) {
      StringBuilder rev_str = new StringBuilder(in.nextLine());
      System. out.println( rev_str.reverse ());
    }
  }
}
// Hypothesis 2: not equivalent to the reference
import java.util.Scanner;
public class Main {
  public static void main(String[] args ) {
    Scanner in = new Scanner(System.in);
    while ( in.hasNext())
      System. out.println( in.nextInt() + in.nextInt ());
  }
}
```

Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002.

BLEU

- Fast
- Works for partial code
- Language agnostic
- Similar-looking different code

0.48

```
// Reference:
import java.util.*;
public class Main {
  public static void main(String[] args ) {
    Scanner in = new Scanner(System.in);
    int t = in.nextInt();
    in.nextLine();
    while ( t-- > 0 ) {
      System. out.println( new StringBuffer(in.nextLine()).reverse
      ());
    }
  }
}
// Hypothesis 1: equivalent to the reference
import java.util.Scanner;
public class Main {
  public static void main(String argv[] ) {
    int num_of_tests = 0;
    Scanner in = new Scanner(System.in);
    num_of_tests= Integer.parseInt(in.nextLine());
    for(int i=0; i<num_of_tests; i++ ) {
      StringBuilder rev_str = new StringBuilder(in.nextLine());
      System. out.println( rev_str.reverse ());
    }
  }
}
// Hypothesis 2: not equivalent to the reference
import java.util.Scanner;
public class Main {
  public static void main(String[] args ) {
    Scanner in = new Scanner(System.in);
    while ( in.hasNext())
      System. out.println( in.nextInt() + in.nextInt ());
  }
}
```

0.55

Difference between PL & NL

Syntax, coding conventions, etc.

→ Common n-grams

→ Trivially shared n-grams

```
while (iterator.hasNext()) {
    final TaskDescriptor taskDescriptor = iterator.next();
    if (taskDescriptor.isHaveMagicCure()) continue;
    final Object taskCure = taskDescriptor.hasCure(disaster);
    if (! cure.equals(taskCure)) {
        iterator.remove();
    }
}
```

```
for (File file : File.listRoots()) {
    String rootPath = file.getAbsolutePath();
    String normalisedStr = notationString;
    if (!fileSystem.isCaseSensitive()) {
        rootPath = rootPath.toLowerCase();
        normalisedStr = normalisedStr.toLowerCase();
    }
}
```

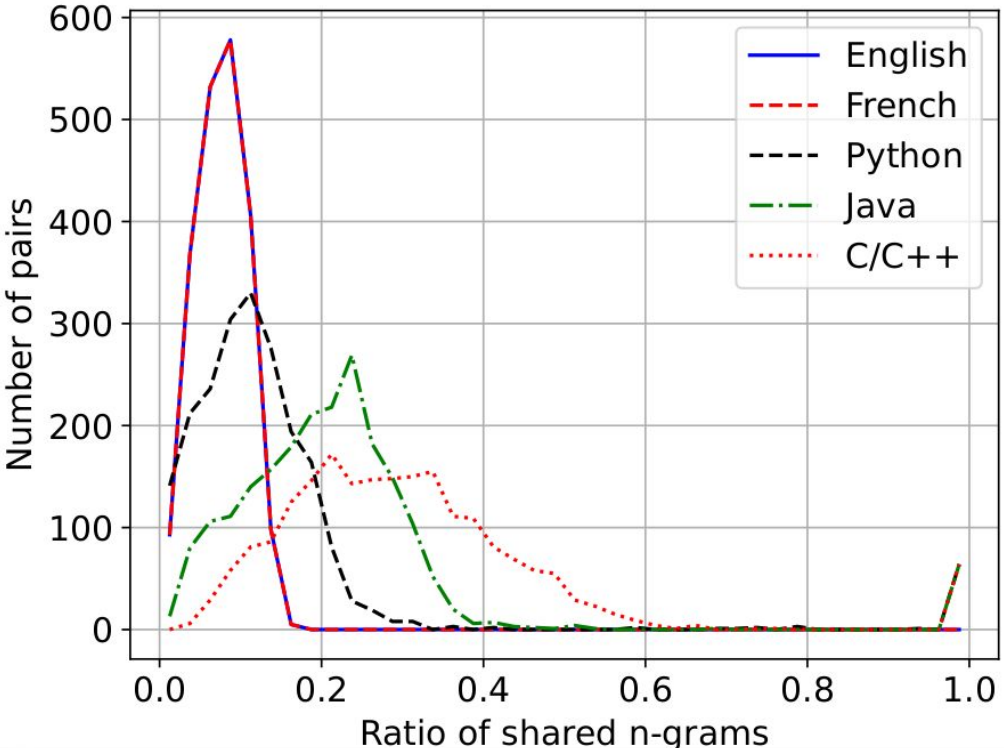
Most Common N-grams

Most frequent n-grams in PL are more frequent that the most frequent n-grams in NL

	2-grams	% of 2-grams	4-grams	% of 4-grams
Java	) ;	5.49	) ; } }	1.34
	( )	4.75	( ) { return	1.29
	) {	3.83	( ) ; }	1.14
	; }	3.81	)) ; }	1.12
	; import	2.75	) ; } public	1.00
	))	1.73	( )) ;	0.94
	} public	1.27	) { this .	0.68
	{ return	1.24	; } public void	0.61
	} }	1.07	; } @Override public	0.54
	) .	0.99	) { if (	0.54
English	of the	2.31	” , he said	0.56
	, and	1.45	, he said ,	0.32
	in the	1.31	, of course ,	0.31
	” .	1.01	” , I said	0.29
	, the	0.85	” , she said	0.29
	to the	0.85	, he said .	0.27
	. “	0.64	” . “ I	0.22
	” ,	0.63	he said , “	0.21
	on the	0.57	” ? asked .	0.19
	, but	0.53	, I said .	0.19

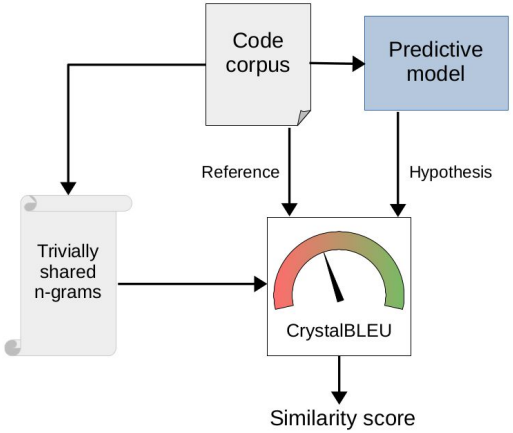
Unrelated Pairs of Text

Two PL texts share more n-grams than two NL texts



CrystalBLEU

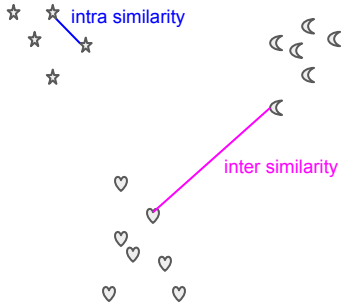
- Extract common n-grams (as trivially shared n-grams)
- Run BLEU, ignoring the top K (~500) common n-grams
- Similar to stop words in NLP



Function *modified_precision(ref, hyp, i, S)* **is**
  refCounts ← n-grams of length *i* from each *ref* and their number of occurrences
  hypCount ← n-grams of length *i* from *hyp* and their number of occurrences
  remove any n-grams from *refCounts* and *hypCount* that is in *S*, or divide *refCounts* and *hypCount* by the logarithm of counts in *S*
  for *ngram* ∈ *hypCount* **do**
    $clipped\_count_{ngram} \leftarrow min(hypCount_{ngram}, max(refCounts_{ngram}))$
  end
  $numerator \leftarrow \sum_i clipped\_count_i$
  $denominator \leftarrow max(1, \sum hypCount)$
  return *numerator, denominator*
end

Distinguishability

Measure how well a metric can distinguish similar and dissimilar pairs



Ratio of similarity within classes to between classes

$$d = \frac{m(Pairs_{intra})}{m(Pairs_{inter})}$$

Research Questions

RQ1: How well does CrystalBLEU distinguish similar and dissimilar programs?

RQ2: Can CrystalBLEU avoid misleading comparisons?

RQ3: How efficient is calculating CrystalBLEU?

RQ1: Distinguishability

ShareCode online judge

Human written code

Accepted solutions to a problem → Class of equivalent programs

	BLEU	CodeBLEU	CrystalBLEU
Intra-class	0.79	0.52	0.65
Inter-class	0.32	0.36	0.10
Distinguishability	2.47	1.44	**6.50**

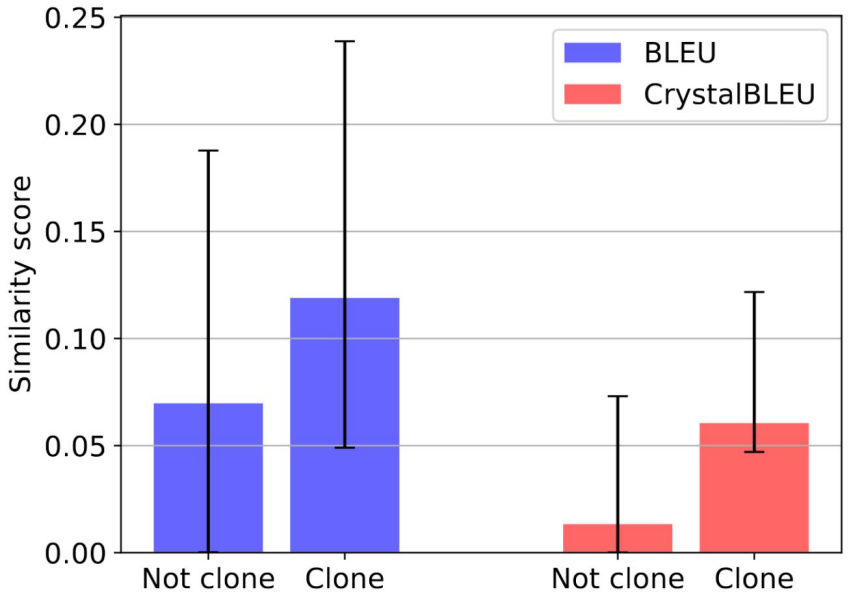
https://sharecode.io/

RQ1: Clone Detection

BigCloneBench

Simple threshold-based classification

	BLEU	CrystalBLEU
Accuracy	0.66	**0.82**
Precision	0.20	**0.37**
Recall	**0.52**	0.46
F1 score	0.20	**0.37**



Svajlenko, Jeffrey, et al. "Towards a big data curated benchmark of inter-project code clones." *2014 IEEE International Conference on Software Maintenance and Evolution*. IEEE, 2014.
Wang, Wenhan, et al. "Detecting code clones with graph neural network and flow-augmented abstract syntax tree." *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2020.

RQ2: Evaluating Code Generators

Ground truth

```
Rational function ( int arg0 ) { VideoTrack loc0 = tracks . get ( arg0 ) ; return new Rational ( loc0 . count , 30 ) ; }
int function ( ) { return this . leased ; }
void function ( ) { SecurityConfiguration . getApplicationPolicy ( "srini_string" ) ; }
boolean function ( ) { return ( type == DICTIONARY ) ; }
void function ( List < Integer > arg0 ) { takeLock . lock ( ) ; try { taskIdsQueue . add ( arg0 ) ; notEmpty . signal ( ) ; } finally { takeLock . unlock ( ) ; } }
void function ( boolean arg0 ) { fResolveBindings = arg0 ; }
boolean function ( ) { return ui . findAll ( locator ) . length == 1 ; }
boolean function ( ) { return ( type == DICTIONARY ) ; }
void function ( int arg0 ) { this . level = arg0 ; }
void function ( int arg0 ) { int [ ] loc0 = extractKeys ( arg0 ) ; for ( int loc1 = loc0 . length - 1 ; loc1 >= 0 ; -- loc1 ) doKeyUp ( loc0 [ loc1 ] ) ; }
```

Neural model

```
---------------------------------------------------------------
int function ( ) { return 0 ; }
int function ( ) { return lease ; }
void function ( ) { SecurityConfiguration . getApplicationPolicy ( ) ; }
boolean function ( ) { return type == NAME ; }
void function ( List < List < Integer >> arg0 ) { taskIdsQueue . addAll ( arg0 ) ; }
void function ( boolean arg0 ) { fBindingsRecovery = arg0 ; }
boolean function ( ) { return locator != null ; }
boolean function ( ) { return type == NAME ; }
void function ( int arg0 ) { this . level = arg0 ; }
void function ( KeyEvent arg0 ) { }
```

Dummy model

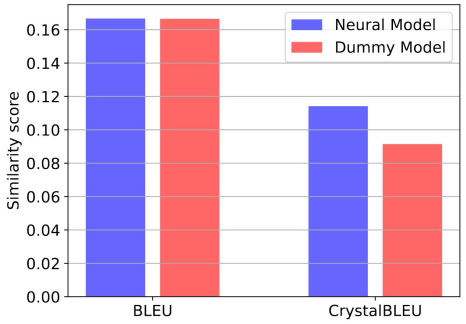
```
---------------------------------------------------------------
= ) ; } loc0 arg0 ) , return ( ) ) ; function ( function ; } ) { } { arg0 . ; ) ( function (
; } ) { } { arg0 . ; ) (
function ( function ; } ) { } { arg0 . ; ) (
function ; } ) { } { arg0 . ; ) ( function
function ( ) { function ( ) ( ) { ) { return { return = ) ; } loc0 arg0 ) , return ( ) ) ; function ( function ; } ) { } { arg0 . ; ) ( function ( List
; } ) { } { arg0 . ; ) ( function
) ; function ( function ; } ) { } { arg0 . ; ) ( function (
function ; } ) { } { arg0 . ; ) ( function
function ( function ; } ) { } { arg0 . ; ) ( function
function ( ) { function ( ) ( ) { ) { return { return = ) ; } loc0 arg0 ) , return ( ) ) ; function ( function ; } ) { } { arg0 . ; ) ( function ( int
```

Neural Model:

Concode*

Dummy Model:

Generate code mostly consisting of common n-grams



* Iyer, Srinivasan, et al. "Mapping language to code in programmatic context." *arXiv preprint arXiv:1808.09588* (2018)

CrystalBLEU

```
// Reference:
import java.util.*;
public class Main {
  public static void main(String[] args ) {
    Scanner in = new Scanner(System.in);
    int t = in.nextInt();
    in.nextLine();
    while ( t-- > 0 ) {
      System. out.println( new StringBuffer(in.nextLine()).reverse
      ());
    }
  }
}
// Hypothesis 1: equivalent to the reference
import java.util.Scanner;
public class Main {
  public static void main(String argv[] ) {
    int num_of_tests = 0;
    Scanner in = new Scanner(System.in);
    num_of_tests= Integer.parseInt(in.nextLine());
    for(int i=0; i<num_of_tests; i++ ) {
      StringBuilder rev_str = new StringBuilder(in.nextLine());
      System. out.println( rev_str.reverse ());
    }
  }
}
// Hypothesis 2: not equivalent to the reference
import java.util.Scanner;
public class Main {
  public static void main(String[] args ) {
    Scanner in = new Scanner(System.in);
    while ( in.hasNext())
      System. out.println( in.nextInt() + in.nextInt ());
  }
}
```

0.24

0.00

RQ3: Runtime

Preprocessing

Dataset	# of tokens	Preprocessing time (s)
ShareCode Java	580K	4.8
ShareCode C++	1.8M	19.9
BigCloneBench	2.6M	22.3
Concode (tokenized)	2.6M	4.1

Metric calculation (100 pairs)

	BLEU	CodeBLEU	CrystalBLEU
ShareCode Java Intra	1036.9	5382.3	**953.6**
ShareCode Java Inter	868.9	3848.3	**743.6**
BigCloneBench Intra	**83.5**	1445.1	85.7
BigCloneBench Inter	**81.5**	1269.4	81.7

How to Use

Install:

```
> pip install crystalbleu
```

Use:

```
from collections import Counter
# Import CrystalBLEU
from crystalbleu import corpus_bleu

# Extract trivially shared n-grams
k = 500
frequencies = Counter(tokenized_corpus) # tokenized_corpus is a
                                        # list of strings
trivially_shared_ngrams = dict(frequencies.most_common(k))

# Calculate CrystalBLEU
crystalBLEU_score = corpus_bleu(
    references, candidates, ignoring=trivially_shared_ngrams)
```

CrystalBLEU Features

- As fast as BLEU
  - Faster than CodeBLEU
- Works on partial code
- Language agnostic
- Higher distinguishability

Property	BLEU	CodeBLEU	RUBY	CrystalBLEU
Language-agnostic	✅	❌	❌	✅
Handle incomplete and partially incorrect code	✅	❌	❌	✅
Efficient	✅	❌	❌	✅
High distinguishability	❌	❌	N/A	✅

Q&A

CrystalBLEU

Install:

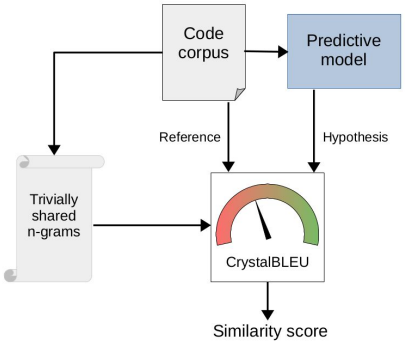
```
> pip install crystalbleu
```

Use:

```
from collections import Counter
# Import CrystalBLEU
from crystalbleu import corpus_bleu

# Extract trivially shared n-grams
k = 500
frequencies = Counter(tokenized_corpus) # tokenized_corpus is a
                                        # list of strings
trivially_shared_ngrams = dict(frequencies.most_common(k))

# Calculate CrystalBLEU
crystalBLEU_score = corpus_bleu(
    references, candidates, ignoring=trivially_shared_ngrams)
```



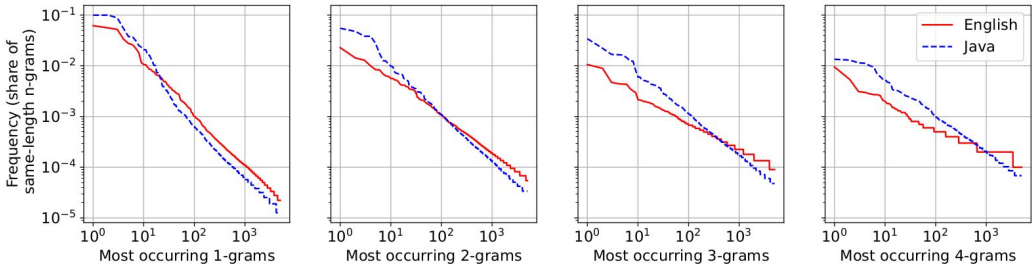
Property	BLEU	CodeBLEU	RUBY	CrystalBLEU
Language-agnostic	✅	❌	❌	✅
Handle incomplete and partially incorrect code	✅	❌	❌	✅
Efficient	✅	❌	❌	✅
High distinguishability	❌	❌	N/A	✅

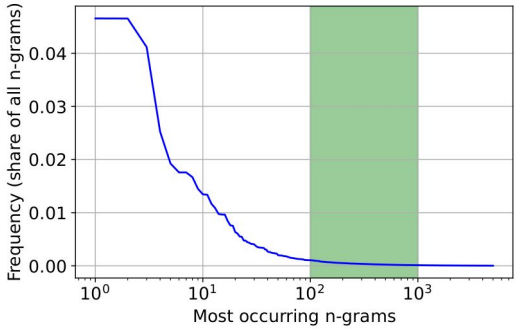
Aryaz Eghbali (aryaz.egh@gmail.com)

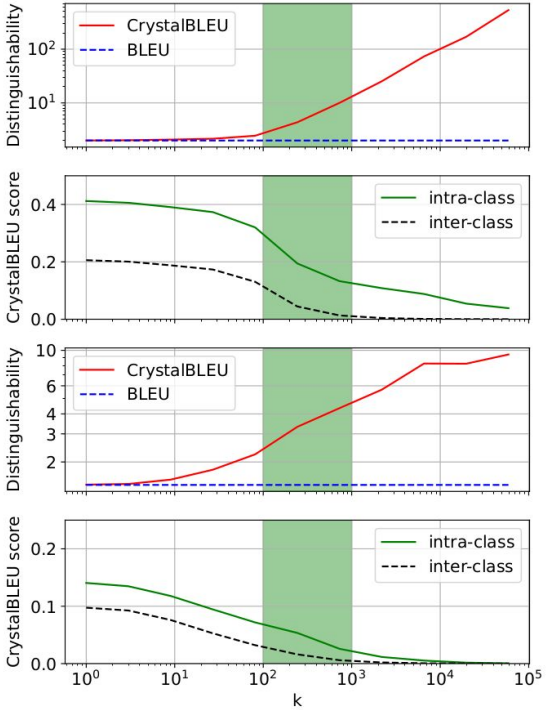
Software Lab (https://software-lab.org/)

Code @ https://github.com/sola-st/crystalbleu

RQ4: Parameter K







Datasets

- English: Brown
- French: Europarl
- Java: Java-small
- Python: py150k
- C++: POJ-104
- Equivalent by behavior: ShareCode
- Equivalent by label: BigCloneBench
- Neural model predictions: Concode

Features

Small human study (1 subject)

Property	BLEU	CodeBLEU	RUBY	CrystalBLEU
Language-agnostic	✅	❌	❌	✅
Handle incomplete and partially incorrect code	✅	❌	❌	✅
Efficient	✅	❌	❌	✅
Correlate well with human judgment	✅	✅	✅	✅
High distinguishability	❌	❌	N/A	✅