社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

畳み込みニューラルネットワークを用いたコード片からのプログラミング言語識別

田中健太郎† 水野 修††

† 京都工芸繊維大学 大学院工芸科学研究科 情報工学専攻
†† 京都工芸繊維大学 情報工学・人間科学系
E-mail: †k-tanaka@se.is.kit.ac.jp, ††o-mizuno@kit.ac.jp

あらまし プログラミング技術に関するナレッジコミュニティとして最も有名なものの一つに Stack Overflow がある．ここでは，プログラミングに関する質問が投稿されると，それに対して回答が寄せられる．投稿される質問や解答にはしばしば snippet と呼ばれるテキスト片が含まれている．このテキスト片は一般的にはソースコードの断片であり，ソフトウェアにおけるバグやその修正を分析する上で，非常に有用なデータセットとなりうる．しかし，snippet には目的のプログラミング言語以外で記述されたテキストもしばしば含まれているため，そのままでは正確なデータとして用いることができない．この問題を解決するためには，「snippet がソースコードであるかどうか」，そして，「どのプログラミング言語によって記述されたソースコードであるかどうか」を識別する必要がある．本報告では，これらの snippet に対して，畳み込みニューラルネットワークを用いたプログラミング言語の識別方法を提案した．Stack Overflow から取得できるオープンデータを用いた実験より，プログラミング言語の分類，及び，snippet から目的のプログラミング言語以外の snippet の除外が可能であることを確認した．

キーワード プログラミング言語識別, Stack Overflow, 畳み込みニューラルネットワーク

Identifying Programming Languages from Code Snippets Using Convolutional Neural Network

Kentaro TANAKA† and Osamu MIZUNO††

† Graduate School of Science and Technology, Kyoto Institute of Technology
†† Faculty of Information and Human Sciences, Kyoto Institute of Technology
E-mail: †k-tanaka@se.is.kit.ac.jp, ††o-mizuno@kit.ac.jp

Abstract Stack Overflow is one of the most famous Q&A website for programming questions. There is a large amount of code snippets in questions and answers and they can be used as an important dataset. However, some snippets are not written in valid programming language (e.g. in natural language), so it cannot be used as an accurate dataset. In order to solve this problem, we have to identify whether the snippets contain programming language or not and by which programming language the snippets are written. In this paper, we propose a programming language identifier using a convolutional neural network. The empirical experiment shows that our method successfully distinguishs 9 difference programming languages.

Key words Programming Language Identification, Stack Overflow, Convolutional Neural Network

1. はじめに

プログラミング技術に関するナレッジコミュニティとして最も有名なものの一つに Stack Overflow がある [1]．ここでは，プログラミングに関する質問が投稿されると，それに対して回答が寄せられる．投稿される質問や解答にはしばしば snippet と呼ばれるテキスト片が含まれている．この snippet は一般的にはソースコードの断片であり，ソフトウェアにおけるバグやその修正を分析する上で，非常に有用なデータセットとなりうる．しかし，snippet には目的のプログラミング言語以外で記述されたテキストもしばしば含まれているため，そのままでは正確なデータとして用いることができない．この問題を解決するためには，「snippet がソースコードであるかどうか」，そして，「どのプログラミング言語によって記述されたソースコードである

かどうか」を識別する必要がある．

既存のプログラミング言語識別手法として Klein らの手法がある [2]．この手法はソースコードの文法・語彙からプログラミング言語を推定するものであるが，文字列やコメント部分を推定して削除する処理が含まれている．これは，コンパイル・実行可能なソースコードに対しては有効な手段であるが，Stack Overflow 等に記載されている snippet に対しては，特にプログラミング言語以外が記述されている場合に，意図しない動作をする危険性があり，また，推定精度も上がらないと予想される．そのため，構文解析や語彙に依らない手法が必要となる．これ以前に提案されていた手法として，SourceClassifier [8] がある．しかし，Kelin らは「このツールは単純なベイズ識別器に基づいており，学習データの質に依存し，文字列とコメントによって精度が落ちる」と述べている [2]．また，GitHub によるプログラミング言語識別手法として，Linguist [3] がある．しかし，Linguist はファイルを対象としており，ファイルの拡張子の情報なども用いた識別を行うため，拡張子の存在しない snippet には適用が難しい．一方で，多くのツールがプログラミング言語識別の実装として提案されており [4]，プログラミング言語識別に対する実用的なニーズは高いことが分かる．

そこで我々は機械学習によって snippet の記述言語を推定する手法を提案する．本報告では，snippet がプログラミング言語であるか・どのプログラミング言語によって記述されているかを畳み込みニューラルネットワークを用いて識別する手法の紹介とそのケーススタディについて述べる．

本報告における研究設問 (Research Question: RQ) として，以下の 2 つを設定した．

(RQ1)　Stack Overflow の質問・回答内の snippet から，目的のプログラミング言語以外の snippet を除外できるか?
(RQ2)　Stack Overflow の質問・回答内の snippet が記述されたプログラミング言語を識別できるか?

本報告の構成は以下のようになっている．節 2. において，研究の理論である畳み込みニューラルネットワークについて説明する．節 3. では，実験についての説明を行う．節 4. では RQ1 についての分析，また，節 5. では RQ2 についての分析を述べる．最後に節 7. で，本報告をまとめる．

2. 畳み込みニューラルネットワーク

畳み込みニューラルネットワークはコンピュータビジョンでよく用いられる手法であるが，近年，自然言語処理の分野においても適用され始めている．本報告における畳み込みニューラルネットワークの構造は Kim の手法 [5] に基づいている．以下では畳み込みニューラルネットワークを用いたテキスト分類で行われている処理の内容を，図 1 を用いて簡単に説明する．

画像分類では入力は画像のピクセル列であるが，言語処理では行列で表現された文章が入力となる．行は各語，列は各単語を表現するベクトルである．単語をベクトル空間に落とし込んで表現することを単語埋め込み表現という．図 1 の例では各単語は 6 次元のベクトルで表現されている．文章行列は画像の RGB チャンネルのように複数チャンネル用意されることもあるが，本報告では 1 チャンネルのみを用いる．

言語処理においては，畳み込みは列方向には行わず行方向にのみ行う．つまり，フィルタの幅は入力となる行列の幅と同じに設定する．高さは任意に設定でき，図 1 の例では高さ 3, 4, 5 のフィルタがそれぞれ 2 個ずつ用意されている．学習フェーズでは，解決したいタスクに適応できるようにフィルタの値を自動的に学習される．

文章行列に対して各フィルタの畳み込み計算を行い，各フィルタから特徴ベクトルが出力される．図 1 の例では 6 個のフィルタがあるので，6 個の特徴ベクトルが出力される．

次に，各特徴ベクトルに対してプーリングを適用し，固定サイズの特徴行列を出力します．この処理によって，高さの異なるフィルタを用いても各フィルタから出力される行列が同じサイズになる．プーリングにはいくつかの種類があるが，本報告においては最大プーリングを用いる．これにより，各特徴ベクトルの要素の最大値が出力される．

最後にプーリング適用後の各特徴量を Softmax 関数に入力することで，各クラスに属する確率が出力される．

以上が本研究で利用した畳み込みニューラルネットワークの処理内容である．

3. ケーススタディ

本節では，Stack Overflow における質問，回答内に含まれる snippet から，プログラミング言語であるものとそれ以外の識別とプログラミング言語名の識別を行ったケーススタディについて述べる．ここでは，IEEE Spectrum が 2014 年に発表した [6] 最もよく使われている 9 つのプログラミング言語 C, C++, JAVA, C#, Ruby, Python, JavaScript, PHP, SQL を対象とした識別実験を行う．

3.1 データセット

Stack Overflow を運営している Stack Overflow 社 (旧 Stack Exchange 社) は，コミュニティサイト Stack Overflow に寄せられた投稿のダンプデータを公開している．XML(Extensible Markup Language) 形式で配布されているが [7]，本研究ではリレーショナルデータベースへと形式を変換して使用している．本研究では 2016 年 11 月の時点で配布されていたデータを用いた．

Stack Overflow の主な機能として質問と回答を行う機能，及び複数の 回答の中から最も良い解答を質問者が「承認」する機能が存在する. また質問には，質問内容に関連する単語をタグとして設定する機能があり，多くの場合そこには使用しているプログラミング言語の名前や関連するライブラリの名前が記載されている．

これらの機能を用いて，Stack Overflow のダンプデータから以下の条件に合致する回答に含まれる snippet を取得する．

- 対象とするプログラミング言語の名前が，質問のタグに 1 言語のみ記載されている．

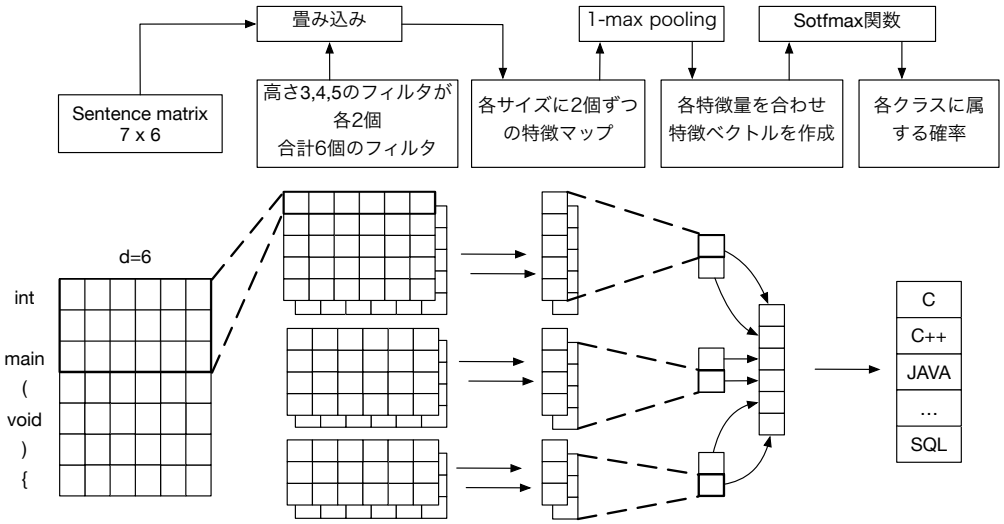

図 1　文章分類のための畳み込みニューラルネットワークの構造

表 1　学習データの LOC とトークン数

タグ	LOC	トークン数
C	81,051	980,552
C++	76,159	868,848
JAVA	79,276	1,052,498
C#	78,161	1,013,083
Ruby	50,387	515,634
Python	61,423	772,000
JavaScript	59,766	797,336
PHP	59,974	725,100
SQL	53,876	733,479
合計	600,073	7,458,530

- 承認された回答である．
- 質問と承認された回答それぞれに，snippet が 1 つずつ存在する．
- 承認された回答に含まれる snippet の行数が 3 行以上である．

snippet は，HTML 言語で保存された各投稿データに <pre>，または，<code> タグで覆われた状態で保存されているため，正規表現を用いて抽出した．

3.1.1　学習データ

学習データには各プログラミング言語毎に，様々な関数・ライブラリを使用しているデータセットを取得する必要がある．しかし，一般のソースコードを幅広く取得することは難しい．

そこで，質問の投稿に付けられたタグを正解ラベルとして，各プログラミング言語のタグごとに 5,000 個，合計 45,000 個の snippet を Stack Overflow から取得し，学習データに用いることにした．

学習データには一部，ソースコード以外の snippet が，正解ラベルを誤った状態で含まれていることに注意しなければならない．学習データの LOC とトークン数を表 1 に示す．

3.1.2　評価データ

評価データは学習データとは別に，各タグごとに 100 個，合計 900 個の snippet を用いた．各 snippet がどのプログラミング言語で記述されているものかを，以下の条件に従って人力で分類をおこなった．

1)　該当の言語のトークン数が全体のトークン数の 60%以下であれば，「その他」に分類する．

2)　言語が C, C++のとき，C++にしか存在しない文法・ヘッダ・メソッドが使用されている場合は C++，使用されていない場合は C と分類する．

3)　プログラミング言語が C, C++以外のとき，該当するプログラミング言語として分類する．

(1) においては，今回はコンパイル・実行可能性については重視していないため，ソースコード以外の記述が一部含まれている場合も，ソースコードとして許容することとした．閾値を 60%と低めの設定しているが，これは今回用意したトークナイザが空白文字をそれぞれ 1 つのトークンとして扱うためである．

(2) については，C は C++のサブセットであるため設定した．

3.2　提案手法

提案手法のプログラミング言語認識手順を以下に示す．

1)　各 snippet を，トークナイザを用いてトークンごとに分割する．

2)　畳み込みニューラルネットワークへ学習データを入力し，学習を行う．タグに記載されているプログラミング言語名を正解ラベルであると仮定し，(1) で得られたトークンの列を学習データとして用いる．

3)　(2) と同様, (1) で得られたトークンの列を畳み込みニューラルネットワークへ入力し，言語の識別を行う．

3.2.1　トークナイズ

畳み込みニューラルネットワークへデータを入力するためには，トークナイザによってテキストを単語 (トークン) ごとに分割しなければならない．しかし，対象となる言語がわからない以上，各言語ごとに正確にトークナイズすることはできない．本実験ではあらゆるプログラミング言語に対応できるように，以下の処理を行うトークナイザを作成した．

1) 英数字及びアンダーバー"_"の連続は一つのトークンとして扱う．(例: token_123)

2) 上記以外の文字は全て，各文字を一つのトークンとして扱う．スペースや改行文字も 1 つのトークンとして扱う．

スペースや改行文字をトークンとして扱っているが，これは Python などの一部の言語ではインデントによってブロックを構成するためである．また，このトークナイザでは文字列やコメントの削除が行われないが，この後の処理において低頻度単語の置換処理が行われるため，その影響は少ないと考えられる．

3.2.2 辞書の作成

学習データのすべての snippet をトークナイズし，各単語の出現回数を調べ，5 回以上出現する単語の辞書を作成する．辞書の肥大化を防ぐため，出現回数が 5 回未満の単語は低頻度単語とし，低頻度単語を示す同一のトークンに変換する．

3.2.3 畳み込みニューラルネットワーク

節 2. において説明した畳み込みニューラルネットワークを用いる，以下に本実験で用いた設定・パラメータを記載する．

- 埋め込み層において，各単語の意味を区別して学習するように埋め込み行列はアトランダムに初期化した．
- 畳み込み層では，高さが 3, 4, 5 の 3 種類のフィルタを用意し，それぞれに 64 個，合計 192 個のフィルタを設定した．
- プーリング層においては，ウィンドウをスライドせず 1 つのフィルタ結果全体に対して最大プーリングを適用した．
- 全結合層では，入力と重み行列の積を取り、バイアスを足す．また，過学習防止のために 0.5 の確率でドロップアプトを実行する．
- 出力層では softmax 関数を通して各クラスに属する確率を出力する．
- 最適化器には Adam optimizer を用いた．
- エポック数は 15, ミニバッチ数は 64 として学習を行った．

softmax 関数を通して各クラスに属する確率が出力されるが，以下の条件で分類を行なった．

- いずれかのクラスに属する確率が閾値以上である場合，最も確率の高いクラスとして分類する．
- 各クラスに属する確率がいずれも閾値以下である場合は，「その他」のクラスに分類する．

4. RQ1 : Stack Overflow の質問・回答内の snippet から，目的のプログラミング言語以外の snippet を除外できるか?

4.1 アプローチ

本研究設問においては，まず，Stack Overflow において収集できる snippet が，その snippet に付けられたタグ（プログラミング言語）と同一であるか否かを判定するために，snippet がある言語のコード片であるか否かの判定法について考える．そのために，節 3.1 で説明したデータセットと節 3.2 で説明した手法を用いて，Stack Overflow の snippet からプログラミング言語以外の除外を行えるかを評価する．

具体的には，各プログラミング言語の評価データセットごとに，表 2 のように 2 値分類問題として評価を行う．

精度 (precision) は以下のように計算することができ，手法適用後の，目的のプログラミング言語が記載された snippet の割合を意味する．

$$\text{precision} = \frac{TP}{TP + FP} \tag{1}$$

再現率 (recall) は以下のように計算することができ，プログラミング言語が記載された snippet の損失割合を意味する．

$$\text{recall} = \frac{TP}{TP + TN} \tag{2}$$

また，データセットに含まれる snippet 中の実際のソースコード片の割合を純度 (purity) と呼ぶことにし，以下のように定義する．

$$\text{purity} = \frac{TP + TN}{TP + FP + TN + FN} \tag{3}$$

精度が評価データの純度よりも高ければ，本手法は有用であると言える．

4.2 結　　果

各タグ毎のプログラミング言語識別結果を表 3 に示す．

評価データの全体の純度は 0.903，精度は 0.975 であり，1%の有意水準で帰無仮説を棄却できるため，本手法によるプログラミング言語以外の snippet の除外は有効であると言える．また，言語毎に見た場合も，全ての言語において，精度 > 純度を満たしており，有効であると言える．

以下では言語ごとの精度・再現率について言及する．

- C++の再現率が低くなっている．コード片を精査したところ，これは一部のソースコードを C と誤認識しているためだと判明した．
- Ruby に関しては，FP の数が 8 と他よりも高い．これは ERB テンプレートを誤って Ruby 言語と学習した影響によるものである．

4.3 RQ1 への回答

Stack Overflow に記載されているタグと本手法を複合的に用いた場合，非常に高い精度で目的の言語が記述されている snippet であるかを識別可能であることがわかった．

5. RQ2 : Stack Overflow の質問・回答内の snippet が記述されたプログラミング言語を識別できるか?

5.1 アプローチ

本研究設問では，多くの snippet の中からプログラミング言語同士を互いに識別できるかを調べる．そのために，節 3.1 で説明したデータセットを用いて，節 3.2 で説明した手法によって，9 つのプログラミング言語をそれぞれ識別できるかどうかを評価する．

5.2 結　　果

プログラミング言語識別結果を表 4 に示す．全体の正答率は 0.847 であった．Klein ら [2] の実験では，全体的な正答率は 0.52 と報告されている．実験の環境が異なるために単純な比較はできないが，彼らの手法よりも高い正答率を示している．

以下では，対象とした言語のソースコードの snippet とそれ

表 2　各タグ毎のプログラミング言語識別

実際	予測	
	目的のプログラミング言語である	目的のプログラミング言語でない
目的のプログラミング言語である	*TP*	*TN*
目的のプログラミング言語でない	*FP*	*FN*

表 3　各タグ毎のプログラミング言語識別結果

タグ	snippet の数	ソースコード数	*TP*	*FP*	再現率	精度	評価データの純度
C	100	94	84	0	0.894	1.000	0.940
C++	100	83	65	1	0.783	0.985	0.830
JAVA	100	92	66	0	0.717	1.000	0.920
C#	100	91	69	2	0.758	0.972	0.910
Ruby	100	89	79	8	0.888	0.908	0.890
Python	100	94	87	1	0.926	0.989	0.940
JavaScript	100	92	87	2	0.946	0.978	0.920
PHP	100	85	78	1	0.918	0.987	0.850
SQL	100	93	87	3	0.935	0.967	0.930
合計	900	813	702	18	0.863	0.975	0.903

表 4　Stack Overflow の回答から取得した snippet に対する，プログラミング言語識別結果

	予測した言語										
実際の言語	C	C++	JAVA	C#	Ruby	Python	JavaScript	PHP	SQL	その他	再現率
C	92	1	0	0	0	0	0	0	0	15	0.852
C++	4	65	0	0	0	0	0	0	0	14	0.783
JAVA	0	0	66	1	0	0	0	0	0	25	0.717
C#	1	0	0	70	0	0	0	0	0	21	0.761
Ruby	0	0	0	0	79	1	1	1	0	7	0.888
Python	0	0	0	0	0	87	0	0	1	6	0.926
JavaScript	0	0	0	1	0	0	87	0	0	4	0.946
PHP	0	0	0	0	0	0	4	78	1	2	0.918
SQL	0	0	0	0	0	0	0	2	94	4	0.940
その他	1	0	0	2	8	1	3	2	4	44	0.677
精度	0.939	0.985	1	0.946	0.908	0.978	0.916	0.940	0.940	0.310	

表 5　識別結果ごとの平均トークン数

	予測した言語									
実際の言語	C	C++	JAVA	C#	Ruby	Python	JavaScript	PHP	SQL	その他
C	241.7	138								135.2
C++	126.5	232.1								171.6
JAVA			272.1	103						169.1
C#	221			199.3						98.4
Ruby					112.5	92	53	117		69.7
Python						213.3			79	74.2
JavaScript				179			187.7			67.8
PHP							87.6	248.5	95	168
SQL								167.5	135.6	125.5
その他	350			262	136.9	166	227.3	47.5	190	217.5

以外の snippet それぞれに対する結果を考察する．

まず，対象とした言語のソースコードの snippet に関しては，いずれも 0.90 以上と高い精度で判別することができた．

Ruby 言語の識別精度が 0.908 と他の言語と比べて低いが，これは Ruby on Rails という Web アプリケーションフレームワークにおける HTML のテンプレート (ERB テンプレート) が Ruby

言語と誤って識別されているためである．誤って識別されるのは，学習データに ruby のソースコード以外にも ERB テンプレートが多分に含まれているためである．このような問題は他の言語においても起きている可能性がある．

また，識別結果ごとの平均トークン数を表 5 に示す．プログラミング言語毎に平均のトークン数は異なるため注意が必要だが，誤認識している snippet は正しく認識できている snippet よりもトークン数が少ない傾向があることが分かる．

その他の snippet に関しては，再現率が 0.677，精度が 0.310 と，対象とした言語のソースコードの結果と比べ再現率・精度共に低い結果となった．精度が低い理由の一つに，トークンの数が少なく識別不能なソースコード片が，その他の snippet に分類されることが挙げられる．そのため，再現率が低い原因として，以下の二つが挙げられる．

- ERB テンプレートが Ruby 言語と誤って識別されている
- その他の snippet として識別する際の閾値が低い

5.3 RQ2 への回答

結果より, 学習対象のプログラミング言語が記述された snippet はそのプログラミング言語が何であるのかを高い精度で識別できることが分かった．一方，その他の snippet を識別するには改善の余地があることも分かった．

6. 妥当性への脅威

この節では，本報告の妥当性への脅威について，構成概念妥当性，内的妥当性，外的妥当性，および信頼性という観点から分析する．

6.1 構成概念妥当性

本実験で用いた学習データには正解ラベルが誤っているものが 1 割程度含まれているため，ERB テンプレートをはじめ，本来意図していない学習が行われている．それらのデータを正しく取り除いた状態で行う実験とは結果が異なることに注意しなければならない．

6.2 内的妥当性

今回の実験ではコメント部分を除き，自然言語を含まないプログラミング言語のみを扱っている．HTML や TeX 等の自然言語が記述されることが前提となっているプログラミング言語を対象とした際にも正しく識別できるかは，別途検証する必要がある．ただし，畳み込み処理の位置不変性を考慮すると，十分に識別可能であると考えられる．

同様に，コメント行が多く含まれるソースコードを正しく分類できるかも別途検証する必要がある．

6.3 外的妥当性

今回の実験で用いたデータセットは Stack Overflow に記載されている snippet に限られている．一般のソースコードに対しても適用可能であるかは GitHub に公開されているソースコード等，より多くの事例に対して適用していく必要がある．

6.4 信 頼 性

実験に用いたデータセットは一般に公開されているものであるため，実験を再現することは容易であると考えられる．

7. 結　　論

本報告では，畳み込みニューラルネットワークを用いた snippet からのプログラミング言語の識別手法を提案した．Stack Overflow から得られた質問・回答データ 45,000 個を用いたケーススタディを実施し，これらに含まれる snippet のプログラミング言語が識別できるかを確かめた．実験の結果，学習対象のプログラミング言語が記述された snippet は高い精度で識別できることを確認した．一方で，それ以外の snippet の識別には改善の余地があることも分かった．

今後の課題として，以下の内容が挙げられる．

- 対象とするプログラミング言語を増やす．
- snippet ではなく一般のソースコードを学習・評価データとして用いて検証を行う．
- 既存のプログラミング言語識別器 [2], [8] との比較を行う．

文　　献

[1] “Stack overflow”. http://stackoverflow.com/

[2] D. Klein, K. Murray, and S. Weber, “Algorithmic programming language identification,” CoRR, vol.abs/1106.4064, 2011. http://arxiv.org/abs/1106.4064

[3] “Linguist”. https://github.com/github/linguist

[4] M.L. Endicott, “100 best github: Language identification”. http://meta-guide.com/software-meta-guide/100-best-github-language-identification

[5] Y. Kim, “Convolutional neural networks for sentence classification,” arXiv preprint arXiv:1408.5882, 2014.

[6] S. Cass, “Top 10 programming languages – spectrum’s 2014 ranking,” IEEE Spectrum, 2014. http://spectrum.ieee.org/computing/software/top-10-programming-languages

[7] “Stack exchange data dump”. https://archive.org/details/stackexchange

[8] C. Lowis, “Identify programming languages with sourceclassifier,” 2009. http://blog.chrislowis.co.uk/2009/01/04/identify-programming-languages-with-source-classifier.html