Empirical Software Engineering (2022) 27: 5
https://doi.org/10.1007/s10664-021-10050-0

Information retrieval versus deep learning approaches for generating traceability links in bilingual projects

Jinfeng Lin[1] · **Yalin Liu**[1] · **Jane Cleland-Huang**[1]

Accepted: 20 September 2021 / Published online: 22 October 2021

Abstract
Software traceability links are established between diverse artifacts of the software development process in order to support tasks such as compliance analysis, safety assurance, and requirements validation. However, practice has shown that it is difficult and costly to create and maintain trace links in non-trivially sized projects. For this reason, many researchers have proposed and evaluated automated approaches based on information retrieval and deep-learning. Generating trace links automatically can also be challenging – especially in multi-national projects which include artifacts written in multiple languages. The intermingled language use can reduce the efficiency of automated tracing solutions. In this work, we analyze patterns of intermingled language that we observed in several different projects, and then comparatively evaluate different tracing algorithms. These include Information Retrieval techniques, such as the Vector Space Model (VSM), Latent Semantic Indexing (LSI), Latent Dirichlet Allocation (LDA), and various models that combine mono- and cross-lingual word embeddings with the Generative Vector Space Model (GVSM), and a deep-learning approach based on a BERT language model. Our experimental analysis of trace links generated for 14 Chinese-English projects indicates that our MultiLingual TraceBERT approach performed best in large projects with close to 2-times the accuracy of the best IR approach, while the IR-based GVSM with neural machine translation and a monolingual word embedding performed best on small projects.

Keywords Software traceability · Cross-lingual information retrieval · Generalized Vector Space Model · BERT

Communicated by: Georgios Gousios and Sarah Nadi

This article belongs to the Topical Collection: *Mining Software Repositories (MSR)*

✉ Jinfeng Lin
jlin6@nd.edu

Yalin Liu
yliu26@nd.edu

Jane Cleland-Huang
janehuang@nd.edu

[1] University of Notre Dame, Notre Dame, IN, USA

1 Introduction

Software traceability captures associations that exist between requirements, design, code, test cases, and other artifacts (Cleland-Huang et al. 2014; Gotel and Finkelstein 1994). The resulting links are then leveraged to support a wide range of software engineering activities such as compliance analysis, safety assurance, and requirements validation (Gotel et al. 2012; Mäder and Gotel 2012). Even though, traceability is required for certification in many domain (Rempel et al. 2015) it is time-consuming and difficult to achieve and maintain accurate links in practice (Hayes et al. 2006). For this reason, many researchers have leveraged information retrieval techniques to automate the creation and maintenance of trace links (Lucia et al. 2007; Rath et al. 2018; Hayes et al. 2006; Rath et al. 2018; 2018; Hayes et al. 2006; Guo et al. 2017).

Common approaches for automatically generating trace links include the Vector Space Model (VSM) (Hayes et al. 2006), Latent Dirichlet Allocation (LDA) (Asuncion and Taylor 2012), Latent Semantic Indexing (LSI) (Antoniol et al. 2002), heuristic techniques (Guo et al. 2013; Spanoudakis et al. 2004), and more recently deep-learning techniques (Guo et al. 2017). IR approaches typically analyze the words in each artifact, compute the syntactic and/or semantic similarity across artifacts, and then assign a similarity score for each pair of artifacts. However, prior work in this area has assumed that all of the artifacts are written in a single language. With the increasing globalization of software projects, this assumption is not always true, as some organizations with multi-national teams, may create artifacts using terminology from two or more languages. We experienced this challenge in a recent collaboration with an international corporation in which we were asked to help them to establish traceability across a very diverse set of documents. We observed that many artifacts included a mixture of English and Chinese. To understand the prevalence of this problem we performed a search of Open Source repositories for other examples of intermingled language use in issues, requirements, and source code. This search was facilitated through a simple topic search, augmented by a specific language name (e.g., : https://github.com/topics/korean). We observed that the use of multi-lingual language appeared fairly often in projects from countries exhibiting low English Proficiency Index (EPI) scores (EF EPI 2019).

We investigated the impact of multi-lingual language use upon the accuracy of several different trace-link generation algorithms with a primary focus on intermingled bilingual artifacts (IBAs) that included English and Chinese. We focused primarily on these two languages because English is typically used as the base language in multi-national projects, and because our collaborators' project included Chinese terms. While we had previously observed extensive use of intermingled English and Greek in an industry project with a different multi-national company, we were not able to find more than a handful of non-Chinese OSS examples. Our initial experiments therefore focused primarily on English-Chinese projects. This paper extends our paper published at the 2020 Conference on Mining Software Repositories (MSR) and entitled "Traceability Support for Multi-Lingual Software Projects" (Liu et al. 2020). In addition to our previous study of Information Retrieval approaches for tracing across bilingual project artifacts, we also explore the use of deep-learning techniques based on pre-trained language models. We use a bilingual language model to integrate the cross-lingual text comprehension and trace link generation into a single neural network so as to avoid the error introduced by explicit language generation in machine translators.

The remainder of the paper is laid out as follows. Section 2 describes the datasets that we use for experiments described in this paper. Section 3 describes the way intermingled

terms and phrases are used across 14 different Chinese-English projects and identifies commonly occurring usage patterns. Sections 4 and 5 describe and evaluate classic information retrieval tracing algorithms – namely the Vector Space Model (VSM), Latent Dirichlet Allocation (LDA) and Latent Semantic Indexing (LSI), and the Generalized Vector Space Model (GVSM) used in combination with both mono- and cross-lingual word embeddings. We show that utilizing GVSM with monolingual word embeddings and a preprocessing translation is more effective than basic IR approaches; however, a cross-lingual approach eliminates the costs of using an external translation service. Section 6 then introduces a deep-learning approach based on the use of a pre-trained language model, and shows that in larger projects it outperforms the GVSM techniques. Finally, in Sections 7 to 9, we discuss threats to validity, related work, and finally conclude by summarizing our results and discussing their relevance.

2 Experimental Datasets

We formally defined intermingled artifacts as follows. Given a pair of artifact types D with source artifact set A_S and target artifact set A_T, then the source artifact $a_{s_i} \in A_S$ is composed of terms in a vocabulary V, where $V = L_p \cup L_f$, and the target artifacts are similarly composed. Further, L_p and L_f are vocabulary subsets of the primary and foreign language respectively.

For example, Fig. 1 depicts language-intermingled issues and/or source code for Turkish, Portuguese, and Chinese OSS systems. In the first case, the commit message is written in Turkish, while code comments are in English. In the second case, there is a list of projects with their descriptions, some of the descriptions are written in English, some of them are in Portuguese. Finally, in the third case, the code and variable name are all written in English, while the comments in code are all Chinese.

2.1 Datasets

For experimental purposes we searched for OSS containing intermingled languages. All of our datasets include English plus one additional language, which we refer to as the *foreign* language. We established four search criteria (C1-C4) for the OSS systems:

- C1 The Project must contain at least 40 issues and commits in its overall development history.
- C2 Foreign terms constituted at least 1% of the vocabulary.
- C3 Commits were routinely tagged with issue IDs. These tags were then used to generate trace links for evaluation purposes.
- C4 The overall set of projects have a varied number of true links between issues and commits. This enabled us to observe the performance of our model in both large and small projects.

With respect to criterion C1, we initially searched for projects with greater numbers of issues and commits; however, we had difficulty finding sufficient projects that met all of our criteria. Given that we have observed the prevalent use of intermingled language in proprietary systems of two multi-national organizations, we conclude that it is less frequent in OSS than in closed-source systems. As we did not have access to sufficient artifacts in these closed-source system to perform experiments, we rely upon OSS to study this phenomenon.

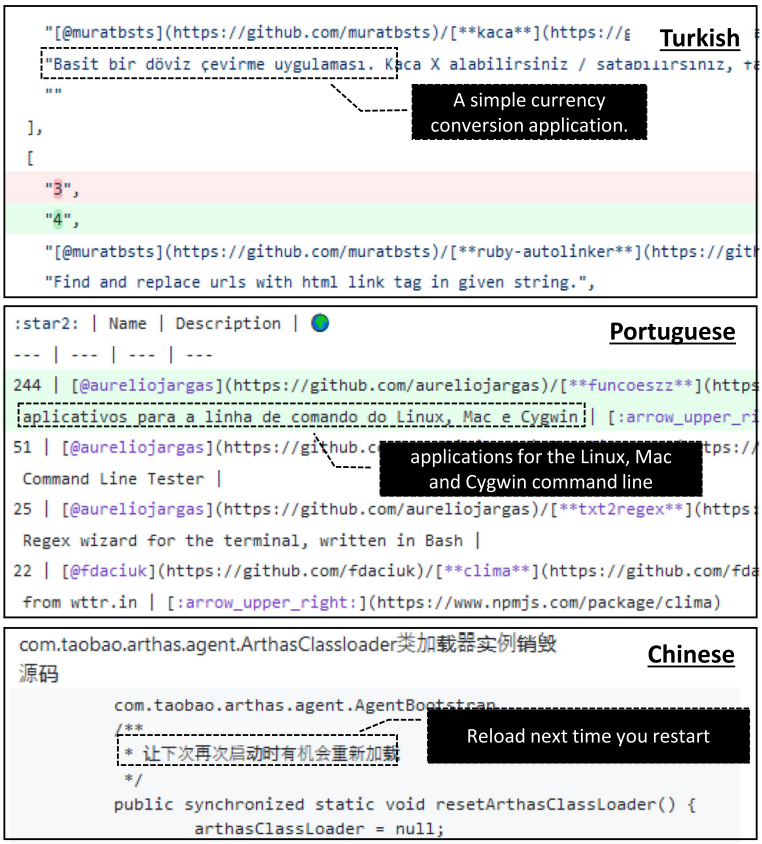


Fig. 1 Three examples of intermingled language used in issues and source code show the use of a second language within an otherwise English language context

We used a systematic process to identify OSS systems that matched our criteria. The process included (1) collecting the names of the biggest Chinese Internet companies based on a survey published by Meeker and Wu (2018). Nine of these Chinese companies were recognized in the top 20 global Internet companies by the market value. Then we started the project search among these companies which included Alibaba, Tencent, Ant Financial, Baidu, Xiaomi, Didi, JD, Meituan, Toutiao. (2) We searched Github for these company names in order to retrieve a list of their OSS repositories, and found enterprise-level repositories for five of the nine companies (excluding JD, Ant Financial, TouTiao and Didi) and added NetEase which is another famous IT company in China. (3) We then sorted the projects in the retrieved repositories according to the numbers of stars to identify the most popular projects. (4) Finally, we selected the top scoring projects that met our defined criteria.

In addition, while our focus throughout this paper is on English-Chinese projects, we also included three additional projects based on Korean, Japanese, and German. However, as large companies in those three countries, e.g. Samsung, Hitachi and Siemens, produce few bilingual projects, we selected three popular personal projects for those languages instead. We searched by the language name, sorted the projects by popularity (star numbers) and

then followed steps (3) and (4) as described above. We discuss results from these additional languages towards the end of the paper. All of the selected projects are depicted in Table 1.

We then used the Github Rest API to parse each of the selected projects and to extract information from commits and issues. We retrieved both the **commit message** and the **source code change set** to establish source artifacts for the trace links. We then collected **issue description**, and **issue summaries** to construct our target artifact sets. We removed all personal identification from the issues, while retaining comments. We then used regular expressions to parse commit messages and extract explicitly defined tags from commits to issues. These tags allowed us to establish a golden-answer set for experimental evaluation. We then used this answer set to evaluate the correctness of candidate links generated by each of our approaches. Candidate links appearing in the answer set were marked as true positives, while others were marked as false positives. An example of a commit and issue is depicted in Table 2, while descriptive statistics for the collected datasets are shown in Table 3a.

While tags in commit messages provide a convenient source of trace links, Rath et al., studied commit messages in five OSS projects and found that an average of 48% were not linked to any issue ID (Rath et al. 2018). This observation means that golden link sets generated from commit tags are unlikely to be complete, and that 'true positive' instances in the evaluation step could be mistakenly treated as 'false positives'. To partially mitigate this problem, we limited the scope of artifacts to those included in at least one of the explicitly specified links from the golden link set. All other artifacts (i.e., issues and commits) were removed. This created a dataset with denser ground-truth link coverage and fewer inaccuracies.

In this study, an artifact is considered as directly impacted if it contains foreign words. Artifacts linked to a directly impacted artifact are regarded as indirectly impacted. To focus

Table 1 OSS projects from Github used in our study description now we have space

Language	Project	Abbreviation	Comp.	Domain
Chinese	Arthas	Ar	Alibaba	Java diagnostics tool for production issues
	bk-cmdb	BK	Tencent	Config. Management system for enterprise level application.
	Canal	Ca	Alibaba	MySQL Database log parser
	Druid	Dr	Alibaba	Database connection pools in JAVA.
	Emmagee	Em	Netease	Performance test tool for Android App
	Nacos	Na	Alibaba	Service discovery and management platform
	NCNN	Nc	Tencent	Neural network library for mobile computing
	Pegasus	Pe	Xiaomi	Distributed key-value storage system
	QMUI_Android	QMA	Tencent	Mobile UI library for Android
	QMUI_IOS	QMI	Tencent	Mobile UI for IOS
	Rax	Ra	Alibaba	React framework for building application
	San	Sa	Baidu	JavaScript component framework
	Weui	We	Tencent	WeChat-like UI framework
	xLua	xL	Tencent	Lua library for integrating C#
Korean	Konlpy	Ko	Personal	NLP package for Korean
Japanese	Cica	Ci	Personal	Font repository for Japanese
German	Aws-berline	Ab	Personal	Web application touring Berlin

Table 2 An example of IBA artifacts

(a) The commit message and its change set served as the *source* artifact. + sign (-sign) refer to added (deleted) content in a commit

Commit ID	2017fb7cf12c...
Commit message	PagerUtils offset的bug, 当0 ti需要修改为0时, 取值不正确
Change set	[-] if (offset > 0) {
	[+] if (offset >= 0) {
	[+] //测试mysql 4
	[+] public void test_mysql_4() throws Exception {
	...

(b) The issue including its description and subsequent discussion served as the *target* artifact

Issue ID	Issue #3428
Summary	缺少java.sql.Time类型适配
Description	- '2019-08-29 13:54:29.999888'这个为啥是6位, 不是3位么?
	- MySQL 5.7 has fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) precision. 因为这里的sql仅仅是用来看看的, 不能拿去数据库执行, 如果要执行的话还得考虑mysql时区与程序时区的问题
	...

In this case, the commit message, issue summary, and commit content all contain foreign terms intermingled with English ones

on the cross-lingual scenarios, we further removed all artifacts that were not impacted, directly or indirectly, by the IBAs. For each link in the golden artifact set, if at least one artifact associated with that link included a foreign language term, then we retained both artifacts and labeled the link as an intermingled link. All other artifacts were removed. In effect, it removed artifacts that were never involved in any intermingled link, and allowed us to focus the evaluation on artifacts and links that were impacted by the use of foreign language terms. Applying these two pruning steps reduced the size of the pruned dataset to an average of approximately 27% of the original issues, 17% of the commits, and 77% of the links as shown in Table 3b [1].

Furthermore, while we could have generated links from all commits to all issues, this artificially deflates the accuracy of the results. We therefore applied a time-based heuristic proposed by Rath et al. (2018) which states that as $Issue_{create} < Commit_{create} < Issue_{close}$, then commits can only be linked to issues that exist (either open or closed) at the time of the commit. Furthermore, as commits are only linked to currently open issues, as closed issues are unlikely to represent a valid trace link.

3 Multilingual Artifacts in Practice

Before evaluating different tracing approaches, we assessed the way in which terms from different languages are intermingled across the issues and commits of English-Chinese bilingual projects and address the following research question:

[1] Our dataset can be found at https://doi.org/10.5281/zenodo.3713256

Table 3 Artifact counts for each dataset as mined from the OSS

Project	Ar	Bk	Ca	Dr	Em	Na	NC	Pe	QMA	QMI	Ra	Sa	We	xL	Ko	Ci	Ab
(a) Artifact counts for each dataset as mined from the OSS																	
Issue	437	1701	1080	2859	106	303	746	254	483	478	846	46	752	520	241	49	107
Commit	489	4504	718	5840	139	471	568	261	296	464	3340	1426	507	741	503	188	299
Links	167	1183	274	1173	32	161	101	163	71	35	573	276	159	53	33	27	75
Foreign	11.0%	7.6%	4.3%	6.7%	19.5	1.0%	29.1%	35.8%	15.5%%	19.4%	8.5%	4.0%	6.0%	30.0%	2.9%	11.0%	14.0%
(b) Artifact counts following pruning to remove artifacts that are not impacted directly, or indirectly, by IBA																	
Issue	122	895	232	1092	31	132	97	160	70	32	560	186	154	52	32	25	74
Commit	167	1178	273	1161	32	161	99	160	71	35	571	275	159	52	33	27	74
Links	167	1179	273	1161	32	161	99	160	71	35	571	275	159	52	33	27	74
Foreign	14.6%	8.3%	5.4%	7.3%	21.9%	1.0%	28.0%	35.3%	16.8%	20.8%	9.0%	8.2%	7.1%	29.5%	7.0%	11.7%	31.0%

– **RQ1:** How are Chinese and English terms intermingled across different artifacts?

3.1 Approach

We used stratified random selection to retrieve 5 issues and 5 commits from each of the first 10 projects listed in Table 1, producing a dataset of 50 issues and 50 commits. We then adopted an inductive open coding technique (Khandkar 2009) to explore the way in which Chinese and English terms were used and intermingled. For each artifact, we identified the primary language (i.e., Chinese or English), marked all phrases written in the secondary language, and then observed the role of those phrases. We then created distinctive tags to represent each of these roles, tagged each issue and commit accordingly, and counted the number of occurrences of each tag. Results are reported in Table 4 annotated with subscripts for Chinese (C) and English (E).

3.2 Observations

By observing instances and combinations of each tag, we gain an understanding of how Chinese and English terms were intermingled in the analyzed artifacts.

- **Term usage in Issues:**
  - 87% of the 50 sampled issues were tagged with ID_C or IA_C, meaning that the majority of issues used Chinese as the primary language.
  - 28 % of the issues were tagged with ID_E or IA_E, meaning that their primary language was English. However, none of these cases included any Chinese words. These numbers do not add up to 100% because one file included a complete translation and was therefore counted as both English and Chinese.

Table 4 Inductive open coding tags for 50 issues and 50 commits

Tag	Usage	Examples	Eng	Ch
(a) Tags used to label the dominant language of the artifacts				
ID	Issue summary	Primary language of issue summary	24	76
IA	Issue description	Primary language of issue description	4	98
CM	Commit message	Primary language of commit message	86	14
(b) Tags used to label roles of non-dominant phrases in a dominant language sentence				
CR	Ext. reference	External system e.g., Tomcat, dashboard	56	0
V	Verb usage	Verbs from non-dominant language e.g., kill, debug	9	0
T	Noun usage	Common objects from non-dominant language e.g., demo, thread, timestamp, 资源池 for resource pool	36	6
ER	Errors and traces	Error messages and stack traces	10	7
AC	Acronym	报错 = 报告错误; PR = pull request	9	0
TAG	Tag use	[feature request], 中文说明](README_CN.md)	6	9
CD	Code snippets	println("代码植入成功");	28	19
CC	Code comments	Comments in natural languages	0	68
BD	Bilingual Duplication	Duplicated content written in two languages	2	2

  - Issues written primarily in Chinese (i.e., ID_C and IA_C) included many English terms and phrases as listed in Table 4. 100% of the issues whose primarily language was Chinese contained some English terms.

- **Term usage in Commits:**
  - 86% of the commits were tagged as CM_E, meaning that the primarily language of the commit message was Chinese. Only 14% had English as the primary language.
  - 68% of commits were tagged with CC_C and 38% with CD_C (Code snippet) meaning that Chinese terms frequently were used in code comments and source code. The use of Chinese terms was especially prevalent in database queries (e.g., for specifying SQL query conditions), and in UI elements (e.g., output messages and UI widget labels).
  - For commits with Chinese as the primary language (i.e., CM_C tag), then 58%, 51%, 6% and 6% of them were also tagged with CR_E, T_E, V_E and TAG_E respectively. This indicated that Chinese comments often included intermingled English phrases where CR (i.e., external references) and T (i.e., noun usage) were the most common forms of intermingling.

To summarize, we observed that Chinese sentences tended to include intermingled English phrases in both issues and source code, while English sentences rarely included Chinese phrases. This makes sense as many Chinese IT personnel have working proficiency of the English language, while few non-Chinese personnel are likely to have proficient Chinese language skills. Our observations indicated that the dominant role of a secondary language was to reference components or to use specific terminology. In related work, Pawelka and Juergens (2015), investigated 15 OSS projects in which English was intermingled with other European languages. They observed that source code included identifiers and comments written in a second language. This differs from our observations of Chinese-English projects, in which we only found comments, but not identifiers, written in Chinese or Pinyin (an alphabetic representation of Chinese characters).

4 Basic Information Retrieval Approaches with Translation

To create an initial baseline we first explored the efficacy of three common algorithms for tracing from Issues to Commits in our English-Chinese bilingual projects. Furthermore, in each case we evaluated the results with, and without, the use of a basic translation step. This preliminary research addresses the fundamental question of whether a simple preprocessing translation step is sufficient for tracing artifacts containing intermingled terms through the following research question:

- **RQ2:** To what extent does the use of neural machine translation (NMT) as a preprocessing step improve the accuracy of trace links generated using VSM, LDA and LSI in an IBA dataset?

4.1 Baseline Information Retrieval Algorithms

The Vector Space Model (VSM), Latent Dirichlet Allocation (LDA) and Latent semantic indexing (LSI) are the three most common approaches for generating trace links in monolingual environments (Ali et al. 2013; Lormans and Van Deursen 2006; Asuncion et al.

2010). However, there effectiveness for use in IBA datasets has not been explored. We start by describing each of these common approaches.

4.1.1 Vector Space Model

VSM is a simple technique that computes term similarity between documents, and has been used in many different trace link recovery tasks (Lucia et al. 2007; Hayes et al. 2006). Despite its simplicity it frequently outperforms other techniques (Lohar et al. 2013). VSM represents the vocabulary of discourse as an indexed linear vector of terms, while individual artifacts (e.g., issues, commits, requirements, design) are represented as weighted vectors in this space. Weights are commonly assigned using TF-IDF (Term frequency-inverse document frequency) in which the importance of a term is based on its occurrence and distribution across the text corpus. VSM assumes the Bag of Words (BOW) model in which the ordering of words is not preserved. Let A_S be the collection of source artifacts and A_T the collection of target artifacts then each artifact $a_i \in A_S \cup A_T$ is represented by the terms $\{t_1 ... t_n\}$ it contains regardless of their order. Each artifact a_i is transformed into a numeric format $a_i = \{w_1, w_2, \ldots, w_n\}$ where w_n indicates the TF-IDF score for t_i. The similarity of two artifacts is then estimated by measuring the distance between their vector representations – often by computing the cosine similarity between source and target vectors as follows:

$$Similarity(a_i, a_j) = \frac{a_i^T \cdot a_j}{\sqrt{a_i^T \cdot a_i}\sqrt{a_j^T \cdot a_j}} \tag{1}$$

VSM indexes words as atomic units which are orthogonal to each other regardless of their semantic similarity. Therefore, the affinity between two artifacts is evaluated with respect to the use of shared vocabulary weighted inversely according to the commonality of the shared terms. IBA datasets have a richer vocabulary than monolingual datasets as terms are derived from two different languages. This increases the likelihood that two semantically similar artifacts are described using different terms, and consequently leads to an underestimation of the artifacts' affinity.

4.1.2 Topic Modeling Approaches

Topic modeling is also frequently used to support automated trace link generation (Asuncion et al. 2010). Topic modeling techniques discover the hidden semantic structures of artifacts as abstract concepts and then represent each artifact as a distribution over the concepts. The most commonly adopted approaches are Latent Dirichlet Allocation (LDA), Latent Semantic Indexing (LSI) and Probabilistic Latent Semantic Indexing (PLSI). LSI, also known as Latent Semantic Analysis (LSA), represents each artifact a_i as a vector of word counts c_n such that each word is represented as $a_i = \{c_1, c_2, ..., c_n\}$. Thus the artifact corpus A can be represented as a matrix $A = \{a_1, a_2, ..., a_m\}$ where m refers to the total number of all artifacts in A. LSI learns the latent topics by applying matrix decomposition, e.g Singular Value Decomposition (SVD) (Lucia et al. 2007). Hofmann et al. proposed a probabilistic variant of LSI which is known as PLSI in 1999 (Hofmann 1999) in which a probabilistic model with latent topics is leveraged as a replacement of SVD. LDA then can be regarded as a Bayesian version of PLSI where dirichlet priors are introduced for the topic distribution. Given the topic distribution vector of source and target artifacts, the affinity between two artifacts can be calculated either with Cosine similarity or with Hellinger distance (Kailath

1967) which quantifies the similarity between two probability distributions. Topic modeling approaches represent each topic by eliciting a group of distinctive words associated with a similar theme, and the topic distribution probability of an individual artifact, represents the affinity of the artifact to that topic. When project artifacts contain foreign languages, the representative topic words are constituted from two (or more) distinct languages. Topic modeling methods have the ability to overcome the challenge of multilingual terms, as topics can be composed of terms from multiple languages. However, the number of words per topic needs to be increased in order to accommodate two or more languages.

4.2 Leveraging Neural Machine Translation

Neural machine translation services, such as Google Translate, are capable of translating documents with complex grammars into diverse languages. Wu et al. demonstrated that, for a specific set of datasets, Google Translate achieved average accuracy equivalent to that of bilingual human translators. Furthermore, current version of Google Translate has addressed approximately 60% of previously known translation errors in popular languages such as English-Chinese, thereby significantly improving performance (Wu et al. 2016). Fu et al. manually compared the performance of seven translation services and concluded that Google Translate provided one of the best English-Chinese translations (Fu 2021). We therefore selected Google translation services for this series of experiments.

4.2.1 Translation as a Preprocessing Step

Our basic approach uses an NMT (Google Translation services) to translate all artifacts in our dataset into mono-lingual (English) representations using a sentence-by-sentence approach. Artifacts were first split into sentences using the pre-trained sentence tokenizer provided by NLTK's PunktSentenceTokenizer (Bird 2006) and regular expressions were then used to identify bilingual sentences. In our case, both English and Chinese used within an artifact are represented with UTF-8 encoding, and regular expressions capture non-English sentences by checking the encoding of their characters. If a character in the sentence has an encoding in the range of CJK Unified Ideographs and its variants (Jenkins 1999), we regard this sentence contain Chinese. Finally, each of the bilingual sentences was processed by Google Translate to generate their English counterparts, which then replaced the original bilingual sentences in their relevant artifacts. As a result, the IBA dataset was transformed into an English mono-lingual dataset.

We also considered the use of token-by-token translation as proposed by Muhr et al. (2010); however, the sentence-level approach allows foreign words to be considered within their context and thereby retains their semantics following the translation. Furthermore, Google Translation Service is trained to handle intermingled terms and phrases within a sentence, and leverages a single NMT model to translate between multiple languages even in cases where a sentence contains intermingled phrases (Johnson et al. 2017). To illustrate this we passed the commit message in Table 2a to Google's sentence level translator which output "PagerUtils offset bug, when offset needs to be modified to 0, the value is incorrect". The same message processed using token level translation produced "PagerUtils offset of bug, when offset need modify for 0 time, value incorrect". In this case, the token level translation distorted the sentence semantics as it translated the Chinese phrases without fully understanding their context. Sentence-level translation is also significantly more efficient and cost-effective than document-level translation, as it requires fewer calls to the translation service and reduces the volume of data submitted by removing sentences written

in pure English. This is important as Google Translate charges more money and responds more slowly on a larger text corpus.

4.3 Evaluating NMT as a preprocessing step

We utilized VSM, LSI, and LDA models, to automatically generate trace links for the original IBA artifacts and also for their translated monolingual counterparts. In both cases we applied the time-constraints described in Section 2 to filter the resulting links.

4.3.1 Metrics

Results were evaluated using the average precision (AP) metric which is commonly used for traceability experiments (Shin et al. 2015). AP evaluates the extent to which correct links are returned at the top of a list of ranked links, and is considered more insightful than recall and precision metrics which simply measure whether a link is found within a set of candidate links. AP is calculated as follows:

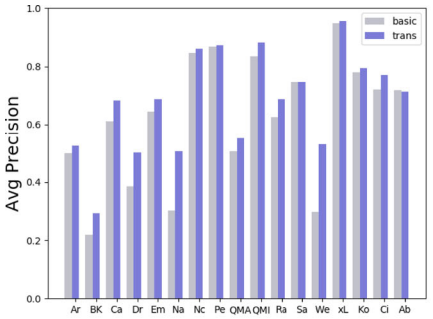
$$AP = \frac{\sum_{i=1}^{n} Precision(i) \times rel(i)}{|true\ links|} \tag{2}$$

where n is the number of candidate links, and rel(i) is an indicator function returning 1 if the link at i_{th} ranking is a true link, otherwise return 0. Precision(i) is the precision score at the cut-off of i_{th} ranking in the returned result. The $|truelinks|$ denominator refers to the total number of true links, meaning that we evaluate average precision for all true links and report AP scores at recall of 100%.

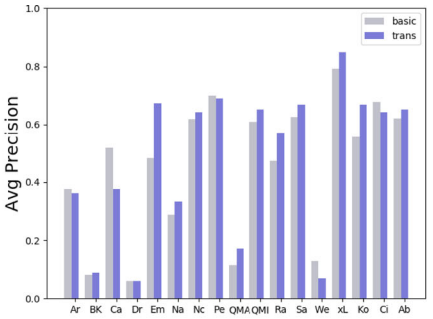
4.3.2 Results and Analysis

To answer RQ2 we compared the AP scores produced for each of the models, with and without Google Translate, for all 14 IBA datasets. The basic models are labeled VSM, LDA and LSI and the corresponding models using NMT are labeled VSM_{tr}, LDA_{tr}, and LSI_{tr}. Project-level results are reported in Fig. 2, and aggregated results across all projects are reported in Fig. 2d. We used the Wilcoxon signed-rank test (Woolson 2007) to evaluate whether the use of translation statistically improved the performance of each technique. This is a standard test used for traceability experiments due to the non-normal distribution of data points. We tested 14 pairs of AP scores achieved from the 14 datasets, with and without translation, using Scipy's (Jones et al. 2001) Wilcoxon test function and adopted the standard significance threshold of 0.05.

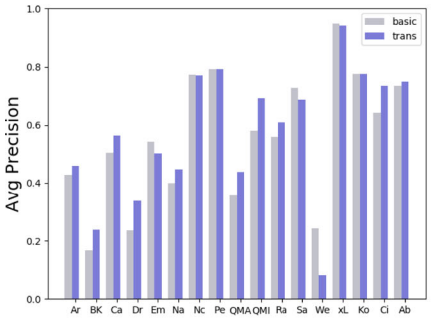
Results showed that VSM_{tr} outperformed VSM with statistical significance (W=2, P = 0.001). On the other hand, in the cases of LSI vs LSI_{tr} and LSI, (W = 34,P = 0.079) and for LDA_{tr} and LDA (W = 43, P = 0.113) there was no statistically significant difference, given that in both cases, the p-values were above the significance threshold. These results indicate that translation improves performance in the case of VSM, but not necessarily for LSI and LDA, most likely because both of these techniques create topic models composed of terms from both languages. As shown in Fig. 2d, both LDA_{tr}, and LSI_{tr} have higher medians, Q1, and Q3 values than their non-translation versions but a lower minimum value. This indicates that in certain cases, translation can degrade the performance of the tracing algorithm instead of improving it. This phenomenon is highlighted in Fig. 2, where we observe that in most projects, the 'trans' version of LDA and LSI have a higher AP score, but there are a few exceptions in which the basic trace models perform better. This result



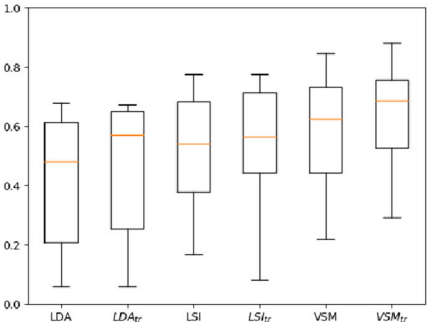
(a) VSM. Average AP score 0.62 (VSM) and 0.68 (VSM_{tr})



(b) LDA. Average AP score 0.45 (LDA) and 0.48 (LDA_{tr})



(c) LSI. Average AP score 0.55 (LSI) and 0.58 (LSI_{tr})



(d) AP score distribution achieved when applying three models for 14 Chinese IBA datasets

Fig. 2 AP scores for three basic trace models, with and without Google Translate, for 17 IBA datasets. Overall best results are observed for the Vector Space Model (VSM)

also confirms previous findings that VSM often outperforms LDA and LSI in various monolingual tracing scenarios (Oliveto et al. 2010; Lohar et al. 2013), our experiment therefore extends these findings to the IBA domain. Given that VSM tends to outperform LSI and LDA on software engineering traceability datasets, it is particularly significant that adding the translation step to VSM (i.e.,VSM_{tr}) provides additional improvements. These results further imply that the presence of bilingual artifacts has a negative effect on traceability results.

4.3.3 Translation Related Pitfalls

We carefully analyzed the outcome of using NMT translation as a preprocessing step to the three commonly used tracing algorithms, and observed the following three failure cases in which translation negatively impacted the results.

Case #1 - Inconsistent Term Translation This occurs when a single Chinese term occurs in both the source and target artifacts but is translated into different English terms due to differing contexts. For example, in one case, a foreign term '启动' appeared as a Chinese verb written in both the source and target artifacts, which resulted in similarity scores when trace links were generated directly for the IBA artifacts. However, NMT translated this term to 'start' in the source artifact and as 'startup' in the target artifact. Neither VSM, LDA,

or LSI captured the semantic similarity between these terms in the translated artifacts. As reported in our earlier study of usage patterns, English sentences seldom contain intermingled Chinese terms; therefore we primarily observed this scenario when tracing between pairs of artifacts written in Chinese such as issues and code comments.

Case #2 - Loss of Meaning This occurs when a relatively specific term in Chinese, is translated into a common English term, thereby introducing noise and subsequently increasing the number of false positive links. As summarized in RQ1, commits with CD_C (Code snippet) tags may contain Chinese terms representing SQL query keywords or UI widget labels. Although these terms are usually sparse with respect to the size of the source code, they serve as strong indicators of true links in the original IBA data when directly referenced in an issue discussion. For example, in the bk-cmdb project, one commit included a JSON file involving the attribute "TopModuleName: 蓝鲸#空闲机池", where "蓝鲸" is a platform name, and '蓝鲸' refers to a pool of machines which are in the IDLE state. NMT translated this clause into 'blue whale free machine pool'. In this case, the unique name '空闲机池' was translated into two common English terms 'blue' and 'whale', introducing the likelihood that the term 'blue' increases the similarity to unrelated issues such as discussions about the UI color schema. At the same time, '空闲' in Chinese is interchangeable with the term 'IDLE', while 'free' has multiple English meanings. As a result, the translation step lost part of the original meaning resulting in a decrease in trace accuracy.

Case #3 - Loss of Reference Phrases Unique phrases that describe specific features in Chinese, are eliminated or weakened by the translation step. We observed that artifact creators appeared to deliberately reference Chinese language content from other artifacts as signature features, indicating that two artifacts are closely related. Translation may inadvertently weaken the association by translating distinctive Chinese terms into common English words, some of which might even be removed as common stop words. This scenario was observed in several artifacts tagged as primarily Chinese (i.e., TAG_C). For example, a tag containing '使用数', which actually means 'how many times it has been used' in English, was translated into the single English term 'number' following the two preprocessing steps of stop word removal (with a word list provided by Many-Stop-Words library 2021) and machine translation. As a result, the true meaning of '使用数' was lost.

4.4 Analysis of Results

Despite these limitations, our results show that adding a translation step into the tracing workflow generally improved the accuracy of the generated links with statistical significance as previously discussed. However, this approach also fails in certain cases when semantic precision or contextual information is lost. From a semantic perspective, NMT can impair tracing accuracy when similar terms are replaced with less similar ones.

To address this problem we therefore explored two different techniques designed to take semantic information into consideration. The first approach leverages word embeddings as a semantic layer to enable those terms to be mapped closer together in the multi-dimensional space, while the second approach uses deep learning based on a pre-trained language model to directly interpret the multilingual text and bypass the translation step. We describe and evaluate each of these approaches in the following two sections.

5 Semantic Information Retrieval with Generalized Vector Space Models

Our first semantically-focused approach explores the use of the Generalized Vector Space Model (GVSM) combined with mono-/cross-lingual word embeddings to address the research question:

— **RQ3:** Which form of word embedding delivers the greatest accuracy when used in conjunction with GVSM to generate trace links in IBA datasets?

After that we compare the efficacy of the GVSM models with the basic Information Retrieval ones to address the following research question:

– **RQ4:** Which of the presented information retrieval tracing techniques (including both the basic and GVSM approaches) generates the most accurate trace links for the IBA datasets?

5.1 Brief Overview of GVSM

GVSM (Wong et al. 1985) is a generalization of the traditional VSM, designed specifically for information retrieval tasks. One of the known weaknesses of VSM is that terms only contribute to the similarity score if they appear in both source and target artifacts. GVSM directly addresses this problem by creating an additional layer of latent space, in which the distance between the terms is determined by their semantic similarity. Given a collection of artifacts $a_i \in A_S \cup A_T$, a_i is vectorized using the standard VSM approach such that $a_i = \{w_1, w_2, \ldots, w_n\}$ where w_n are the weights for the terms in artifact a_i. Considering the vocabulary $V = \{t_1, t_2, \ldots, t_N\}$ composed by all the terms in artifacts, the pairwise term-correlation can be represented as a correlation matrix G of $N \times N$ shape, where $sim(t_i, t_j)$ is the semantic relevance for term t_i and t_j

$$G = \begin{vmatrix} sim(t_1, t_1) & sim(t_1, t_2) & \cdots \\ \vdots & \ddots & \\ sim(t_N, t_1) & & sim(t_N, t_N) \end{vmatrix} \tag{3}$$

In GVSM the similarity between two artifacts is then calculated as follow:

$$Similarity(a_i, a_j) = \frac{a_i^T \cdot G \cdot a_j}{\sqrt{a_i^T \cdot G \cdot a_i}\sqrt{a_j^T \cdot G \cdot a_j}} \tag{4}$$

GVSM has been effectively used to support several different bilingual tasks related to document clustering (Tang et al. 2011), query processing (Wong et al. 1989), and text retrieval (Tsatsaronis and Panagiotopoulou 2009). As shown in (3), GVSM needs a correlation matrix to compute the similarity between words. Researchers have previously leveraged word correlation information (Wong et al. 1985), manually created word thesauri (Hayes et al. 2006; Tsatsaronis et al. 2010), or used concept models (Liu et al. 2020) to accomplish this goal. We opted to use word embeddings which are readily available and require no manual effort to create.

5.2 Integrating Word Embedding (WE) into GVSM

In VSM terms are represented as vectors in a high dimensional orthogonal space in which each term is a dimension of the vector space. In contrast, in GVSM, word embeddings transform the word representation from a high dimensional orthogonal space to a low dimensional non-orthogonal space. The distribution of vectors within the space varies according to the specific approach taken. The use of word embeddings has achieved significant success for addressing NLP challenges in domains such as ad-hoc information retrieval (Almasri et al. 2016; Vulic and Moens 2015), bug localization (Ye et al. 2015), question answering (Dhingra et al. 2016) and also trace link recovery (Zhao et al. 2017; Guo et al. 2017).

We explore two different ways of combining word embeddings into GVSM. The first approach includes a *Cross-Lingual Word Embedding* (CLWE) which projects the vocabulary from two or more distinct languages into a single vector space, thus we name it *Cross-Lingual GVSM* (CLG). The second approach uses an NMT (Google Translation services) to translate the IBA datasets into a monolingual dataset, and then uses a monolingual (English) embedding with GVSM to create trace links. We refer to this approach as *Word Embedding GVSM* (WEG_{tr}). Finally, we create a GVSM baseline for tracing the IBA datasets by omitting the NMT translation step, and refer to this baseline as WEG.

5.3 Cross-lingual Word Embedding (CLWE)

Cross-lingual word embeddings project the vocabulary from two or more distinct languages into a single vector space. A reasonable cross-lingual embedding model organizes term vectors in an embedding space according to their semantic affinities. Cross-lingual embeddings can therefore serve as a semantic bridge between different languages, and can be used to support diverse cross-lingual NLP tasks such as machine translation, cross-lingual IR, and cross-lingual entity linking. Various techniques have been explored for aligning multiple monolingual vector spaces (Ruder et al. 2017). Techniques based on word-level alignment tend to leverage cross-lingual lexicons to supervise the vector mapping. With this approach, monolingual word embeddings are first created for both languages, and then the unseen term vectors for both languages are transformed using a trained mapping model. Researchers have explored other approaches for relaxing word-level alignment constraints, for example by leveraging alignment information at the sentence level (Gouws et al. 2015) or even the document level (Vulic 2017), or entirely foregoing supervision (Wada and Iwata 2018), in order to extend the use of cross-lingual word embedding to additional scenarios.

For our experiments we utilized *relaxed cross-domain similarity local scaling* (RCSLS), which is based on word-level alignment (Joulin et al. 2018). We selected RCSLS for two reasons. First it has been shown to deliver the best overall performance in comparison to other state-of-the-art techniques across 28 different languages (Joulin et al. 2018), and second, pre-trained models with support for 44 languages is available from the FastText library (Fasttext 2021) released by Facebook (Joulin et al. 2018). Facebook trained their model using Wikipedia documents, and leveraged the MUSE library (Conneau et al. 2017) which includes bilingual dictionaries for over 110 languages and monolingual word embeddings for 157 languages including English and Chinese. Vectors in cross-lingual embedding space also have a dimension of 300.

5.3.1 Prior Applications of GVSM for Cross-Lingual Information Retrieval Tasks

Prior work has already investigated the application of GVSM for cross-lingual information retrieval tasks in other domains. For example, Tang et al. (2011) proposed CLGVSM which exploited semantic information from (1) a knowledge base e.g. HowNet (Xia et al., 2011), (2) statistical similarity measures, e.g cosine similarity of term vector covariance (COV), and (3) a bilingual dictionary which contains the translation probability between terms. Other researchers have explored the use of Cross-Lingual Word Embedding to address cross-lingual information retrieval tasks. Vulic and Moens (2015) proposed a model known as cross-lingual information retrieval (CLIR) which directly leverages the distributed representation of cross-lingual vocabulary to accomplish document embedding (DE). Given documents represented by term vectors $d = \{\vec{t_1}, \vec{t_2}, \ldots, \vec{t_n}\}$ where $\vec{t_i}$ is the vector representation of terms, a document vector $\vec{d}$ can be created by merging the term vectors with simple addition. The self-information of the terms (Cover and Thomas 2006), e.g. frequency of terms within a document, can be combined to weight the vectors. The final representation of a document is given as:

$$\vec{d} = w_1\vec{t_1} + w_2\vec{t_2} + \cdots + w_n\vec{t_n} \tag{5}$$

This method, referred to as ADD-SI, projects the documents into the same vector space of terms so that document affinities can be evaluated using distance measures such as cosine similarity. However, we could not find any publications describing the combined use of both GVSM and Cross-Lingual Word Embedding. We replaced the cross-lingual knowledge base in CLGVSM with (Cross-Lingual) Word Embedding, because knowledge tended to be domain-specific and costly to construct for individual software project domains.

As a result, we propose three different techniques for combining GVSM with Word Embeddings.

5.3.2 Application of Cross-Lingual Word Embedding with GVSM (CLG) to Traceability

We utilize a modified cross-lingual word embedding based on GVSM. As shown in (4), a GVSM model is composed of TF-IDF vectors a_i and a semantic relatedness matrix G. The semantic relatedness matrix G can be constructed using external knowledge (Tsatsaronis and Panagiotopoulou 2009; Tang et al. 2011) (e.g HowNet, WordNet) to evaluate term relatedness based on their distance in the knowledge network; or using statistical models that predict the probability of term co-occurrence (Wong et al. 1985). In the first approach, the size of the semantic relatedness matrix is constrained by the vocabulary of the knowledge base. This is a critical limitation for trace link recovery, as software artifacts tend to include a large number of technical terms which are not commonly found in general purpose knowledge sources. Statistical approaches therefore fall far short of capturing the true semantics of these terms. However, these weaknesses can be addressed using word embeddings.

Given an IBA dataset with primary and foreign language vocabulary $L_p = \{t_{p_1}, t_{p_2}, \ldots, t_{p_m}\}$, $L_f = \{t_{f_1}, t_{f_2}, \ldots, t_{f_n}\}$, the monolingual vector for t_{p_i} and t_{f_i} is represented as $x_i, z_i \in \mathbb{R}^d$ where d refers to the dimension of the vector space. As previously discussed, the RCSLC model is capable of projecting vectors from two separate spaces into a shared latent space by learning an internal translation matrix W with the supervision of bilingual lexicons. With this translation matrix W, the vectors can be projected as Wx_i *and* Wz_i. The vocabulary vector space of a given IBA dataset is then represented as:

$$V_S = \{Wx_1, \ldots, Wx_m, Wz_1, \ldots, Wz_n\} \tag{6}$$

As the vectors in RCSLC are $l-2$ normalized (Joulin et al. 2018), the semantic relevance matrix G can be created through the simple dot product of V_S and V_S^T.

The GVSM formula shown as (4) can be transformed into the following:

$$Similarity(a_i, a_j) = \frac{a_i^T \cdot V_S \cdot V_S^T \cdot a_j}{\sqrt{a_i^T \cdot a_i}\sqrt{a_j^T \cdot a_j}} \tag{7}$$

In our case, V_S is the pre-built vector space provided by FastText library as described in Section 5.3

5.4 Using monolingual Neural Word Embeddings with GVSM

Word embeddings are typically used to represent terms from a single language. Their use is based on the intuitive assumption that terms with similar distribution patterns are semantically more similar than those with dissimilar distributions (Harris 1954), leading to the observation that term vectors tend to form clusters according to their semantic affinity. monolingual neural word embeddings (MNWE) leverage the context of each term and include the Skip-Gram model and Continuous Bag Of Words (CBOW) model (Mikolov et al. 2013). While both of these models are built upon simple three-layer neural networks, the Skip-Gram model makes predictions of surrounding terms based on the current terms while the CBOW model predicts a single term using all surrounding terms as context. In our study, we adopt pre-trained monolingual word embeddings that are trained on the Common Crawl dataset with enhanced CBOW model (Mikolov et al. 2018). Vectors in such a space have 300 dimensions.

5.4.1 Transforming CLG to a Monolingual Tracing Task

For experimental purposes we also explored a monolingual version of CLG. The intrinsic difference between monolingual and bilingual trace tasks lies in the dataset vocabulary. As shown in (6), the vocabulary vector space V_s of IBA dataset is composed from two types of vectors 1) term vectors projected from the foreign language space and 2) term vectors projected from English space. For monolingual tracing tasks, the vocabulary vector space $V_{S'}$ contains term vectors of only one language; however, by simply substituting the V_s with $V_{S'}$ in (7) we can migrate CLG to address monolingual tracing tasks. This can be accomplished by training and applying a word embedding model with monolingual text corpus. We name this monolingual model the 'Word Embedding GVSM' (WEG) to distinguish it from CLG. We use this model as our baseline, and then by comparing WEG and GLG we can determine whether the cross-lingual word embedding is able to bridge the semantic gap introduced by the use of two languages in the IBA datasets.

5.4.2 NMT preprocessing with Monolingual Trace Task

As we described above, WEG is the monolingual version of CLG in which cross-lingual word embedding is replaced with an English monolingual embedding. Our third approach combines WEG with NMT to extend its ability to trace IBAs. We followed the same approach used in our initial experiments with VSM_{tr}, LDA_{tr} and LSI_{tr}, by using Google Translation services to translate the IBA datasets back into English monolingual datasets before running WEG. We refer to this method as WEG_{tr} to distinguish it from the other two GVSM models.

5.5 Experimental Evaluation of Semantic Information Retrieval Approaches

All three GVSM models (i.e., CLG, WEG and WEG_{tr}) shown in Table 5 were applied against our experimental datasets. However, due to different amounts of training data available, the size of the cross-lingual embedding tends to be smaller than the monolingual word embedding. To make a fair comparison between the techniques of using monolingual and cross-lingual embeddings, we randomly sampled the vocabulary of the monolingual word embedding to reduce its size. The full monolingual embedding included 2,519,371 records, while the cross-lingual embedding and the down-sized monolingual embedding included only 332,648 records. However, in reality, it is far easier to construct a large monolingual word embedding, and therefore we wanted to see how WEG_{tr} and WEG performed when allowed to use the fully available embedding data. We therefore also include these results, labeled as WEG^*_{tr} and WEG^* respectively. Finally, as an additional point of comparison, we include both VSM and VSM_{tr} from our earlier experiment. Results are reported in Figs. 3 and 4.

5.5.1 RQ3: Analysis of Cross-lingual Word Embedding

To address our research question "Which form of word embedding delivers the greatest accuracy when used in conjunction with GVSM to generate trace links in IBA datasets?" we explored the difference between CLG, WEG, and WEG_{tr} models.

By comparing the average precision achieved for the 14 Chinese datasets as reported in Fig. 3 we observed that in 12 out of 14 cases, WEG_{tr} returned the most accurate trace links. Of the remaining two cases, CLG and WEG each returned top accuracy one time. Furthermore, as reported in Fig. 4, WEG_{tr} had a significantly higher median, Q1, and Q3 value than either monolingual WEG or CLG. Applying full size word embedding to WEG_{tr} further improved the performance by increasing the median value of the results distribution. This indicates that combining WEG with NMT can effectively improve the tracing performance. When comparing WEG and WEG^*, we observe that increasing the embedding size for monolingual WEG had little impact on model performance; however, this contrasts with the marked improvement observed when using an increased English embedding size on WEG_{tr}, reinforcing our conjecture that the vocabulary mismatch introduced by IBA has a clear negative impact upon trace performance.

To determine if it would be possible to avoid the costs of building or contracting a translation service such as Google Translation services, we also compared the monolingual and cross-lingual approaches (i.e., WEG vs. CLG) without the benefit of translation. In this

Table 5 The acronyms and details of the three GVSM and word embedding integrated methods: WE=Monolingual word embedding, CL=Cross-lingual word embedding, TR=Google translation to English

Abbr	GVSM	WE	CL	TR	Description
CLG	■		■		Uses Cross-Lingual word embedding with GVSM. Inputs a bilingual dataset to the model
WEG	■	■			Uses reduced size English Word embedding with GVSM. Inputs a bilingual dataset to the model. WEG^* variant uses full-sized English word embedding.
WEG_{tr}	■	■		■	Uses reduced size English word embedding with GVSM. Uses Google Translate to preprocess IBA data. Inputs resulting mono-lingual dataset to model. WEG^*_{tr} variant uses full-sized English word embedding.

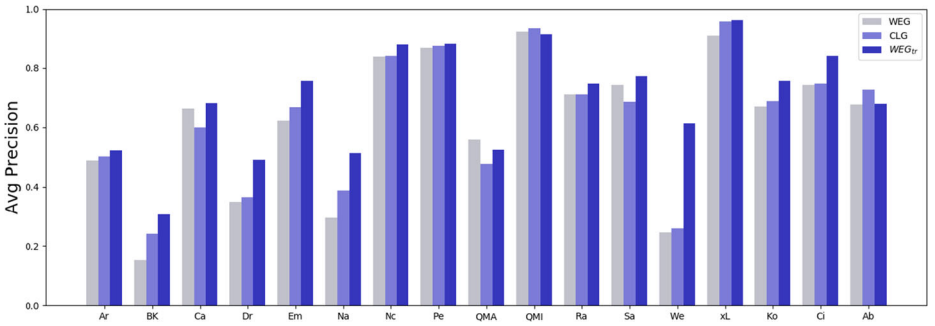


Fig. 3 Average performance of three primary GVSM models for all 17 IBA datasets

case, we observe that CLG outperformed WEG in 10 out of 14 Chinese projects, achieves equivalent performance in one project, and underperforms in 3 projects. However, an analysis of results in Fig. 4 for median, Q1, and Q3 values compared to other models, show that it does not statistically outperform WEG.

We therefore answer RQ3 by stating that the cross-lingual word embedding failed to outperform either of the monolingual word embedding approaches based on the available resources, and that the use of a preprocessing translation step followed by the use of GVSM with monolingual word embedding was clearly superior. One potential reason this model achieved the best performance is that the Google NMT closed the semantic gap between languages better than the cross-lingual word embedding, by taking the context around foreign words into consideration.

5.5.2 RQ4: Comparison of all multilingual IR models

Finally, to address our research question "Which of the presented information retrieval tracing techniques (including both the basic and GVSM approaches) generates the most

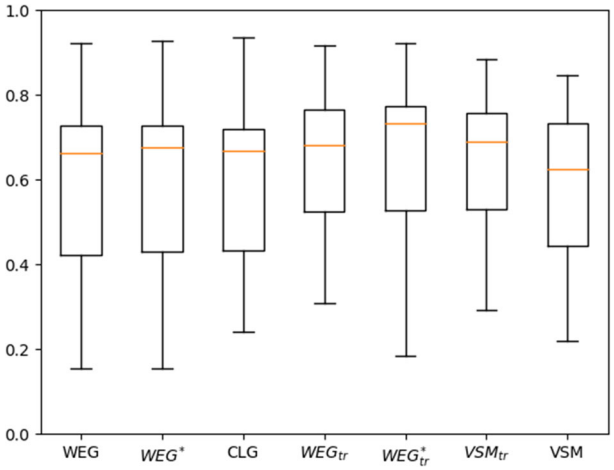


Fig. 4 A comparison of the best basic model (VSM_{tr}) against all three GVSM-based models. WEG and WEG_{tr} with full size English embedding are represented as WEG^* and WEG^*_{tr}

accurate trace links for the IBA datasets?" we compare VSM_{tr}, LDA_{tr}, and LSI_{tr} with our new GVSM-based techniques. As Fig. 2d and Fig. 4 report VSM_{tr} and WEG^{*}_{tr} are observably the best models. We compared AP scores achieved for these two models for all 14 Chinese datasets, against each of the other models using the Wilcoxon signed-rank test and Cohen's d effect size. P-value of Wilcoxon signed-rank test are reported in Table 6, show that both VSM_{tr} and WEG^{*}_{tr} are statistically significant better than other models given the P-values are all below 0.05 with effect size ranging from 0.3 to 0.9 indicating a "medium" to "large" effect. However, a similar comparison of WEG^{*}_{tr} and VSM_{tr} returns a P-value 0.0615 and effect size of 0.09, meaning that neither technique is significantly better than the other even though Fig. 4 shows that WEG^{*}_{tr} has a higher maximal, Q1, Q3 than VSM_{tr}.

5.6 Extension to Other Languages

While our focus was on Chinese-English language projects, we also included one project from each of three additional languages in our experiments as a preliminary proof of concept. These projects were Korean, Japanese, and German – all combined with English. In all cases, including the German language, we were able to use Unicode to identify its presence in English sentences. While we were able to identify language occurrences in our study (i.e., Chinese, Japanese, Korean, and German) from English using Unicode, we will need to adopt more diverse approaches (e.g., Python's langdetect project), for differentiating between more diverse languages. Traceability results for these projects were reported throughout this section of the paper (shown on the right hand side of Table 3, and the graphs of Figures Fig. 3 and Fig. 2d). They show that for both Asian languages (Korean and Japanese) our conclusion derived from the Chinese datasets is still valid, while for the European language (German), the CLG model outperformed WEG_{tr}.

6 Multi-lingual Language Model Based Trace Model

In the previous section, we discussed the combination of IR methods and word embeddings for improving the accuracy of trace link generation; however, the resulting accuracy was still far from ideal for supporting industrial uptake (Cleland-Huang et al. 2014; Cleland-Huang et al. 2014). One of the remaining problems is that multilingual IR trace models do not consider the context of words when calculating similarities between artifacts. Taking CLG as an example, we leveraged a pre-built cross-lingual word embedding to reduce the semantic gap introduced by the multilingual vocabulary. The similarity score between two words

Table 6 P-values of wilcoxon signed-rank test and their effect size for IR models

	WEG^{*}	CLG	LDA_{tr}	LSI_{tr}
(a) P-values of wilcoxon signed-rank test				
WEG^{*}_{tr}	.001	.003	0 .000	.001
VSM_{tr}	.019	.010	0 .000	.001
(b) Cohen's d effect size				
WEG^{*}_{tr}	.325	.325	.941	.671
VSM_{tr}	-.093	.263	.910	.507

was statically calculated based on the distance between their embeddings regardless of the surrounding words in the current context. Furthermore, the multilingual IR trace model is directly impacted by the performance of its constituent parts (i.e., word embeddings and machine translators). For example, in WEG_{tr} any inaccuracies in the monolingual word embedding or the Google machine neural translator are propagated into the generated trace links of a multilingual dataset. Furthermore, as our datasets come from multiple domains, it is difficult to fully calibrate the word embedding and machine translator to the terminology of each domain.

We addressed this problem through leveraging a multilingual Language Model to generate trace links. We refer to this approach as MT-BERT (Multilingual Traceability BERT)[2]. MT-BERT uses BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers) as its underlying language model. We first describe the multilingual language model, and then discuss its use for software traceability. We have previously proposed BERT based traceability models for use in monolingual projects (Double Blinded 2020); here we customize them for addressing the multilingual traceability challenge. To evaluate the effectiveness of this approach, we address the following research questions:

- **RQ5:** Does MT-BERT increase the accuracy of generated trace links in comparison to multilingual IR models?
- **RQ6:** Can we transfer knowledge across projects to increase MT-BERT's effectiveness in smaller projects which lack sufficient training data?

6.1 Multilingual Language Model

A language model is a probability distribution of words in a text corpus. In contrast to word embeddings, the language model is capable of distinguishing the semantic meaning of words based on their surrounding context. For example, the word 'boot' has different meanings according to its current context. When used as a noun it could refer to shoes, whilst its use as a verb could refer to starting up a system. Word embedding approaches can not differentiate between these two meanings, while the language model is capable of inferring the context and true meaning based on the semantics of the entire sentence.

BERT (Devlin et al. 2018) and its variants are widely used language models that support diverse natural language processing tasks. They leverage transformer layers (Vaswani et al. 2017) to overcome the limitations of conventional Recurrent Neural Networks, such as vanishing and exploding gradients which can occur when training large networks. BERT models are able to process longer word sequences and can therefore often achieve better results on specific NLP tasks than RNN models. Researchers train BERT models using Masked Language Modeling (MLM) and Next Sentence Predicting (NSP) tasks (Devlin et al. 2018). In the MLM task, which is of particular relevance to the tracing challenge, a subset of the words in the corpus are replaced with a special '[MASK]' token, and the BERT model is trained to predict the hidden words based on their context.

Multilingual BERT is a special variant of the BERT model trained with 104 common languages (Google-Research 2019). This model was developed and released by Google and has been used in the Cross-lingual Natural Language Inference (XNLI) challenge (Conneau et al. 2018). We adopt this multilingual BERT model within MT-BERT.

[2]Repository for MT-BERT: https://github.com/jinfenglin/EMSE2020

6.2 Multilingual Traceability BERT

Our novel MT-BERT architecture for supporting multilingual software traceability is depicted in Fig. 5. In this architecture, the multilingual language model is utilized as a text encoder and referred to as the 'Multilingual BERT Encoder'. The architecture supports five steps which are used to analyze the semantics of a candidate source and target artifact, and infer their relatedness. First, MT-BERT tokenizes the words in the source and target artifacts. Character-based languages, such as Chinese, Japanese and Korean, are split into characters which are then treated as tokens. Second, these tokens are processed by an embedding layer and converted into embedding representations that are compatible with the neural network in the multilingual BERT encoder. Third, the multilingual BERT encoder processes the embedding representations of source and target artifacts one by one, and generates two vectors conveying the semantic meaning of the artifacts. In the fourth step, these two feature vectors are then concatenated to generate a fused feature vector which is passed in the final step to a simple three-layer feed-forward neural network designed for binary classification. This neural network predicts the affinity score within a range of 0 to 1, that indicates the likelihood that a given pair of artifacts are related.

MT-BERT potentially overcomes the limitations of multilingual IR trace models in two ways. First, by leveraging the multilingual BERT language model to encode the intermingled bilingual artifacts, it avoids the need for machine translation as a prepossessing step. The artifact text is transformed into a context-aware semantic feature vector which can be directly consumed by a neural classifier dedicated to the task of trace link generation. In this way, we eliminate the propagated error caused by machine translation in IR models. Second, pre-existing trace links for a project are used to train MT-BERT by fine-tuning the parameters in both the language model and the neural classifier. When available, these

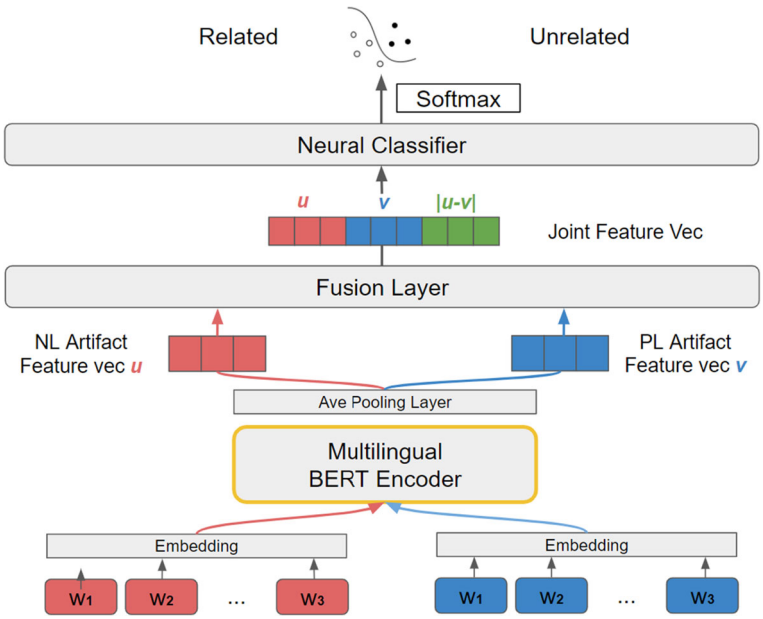


Fig. 5 Architecture of MT-BERT

training examples allow MT-BERT to adapt to the language and traceability needs of a specific project.

However, training examples, in the form of manually created trace links, may not be available in all software engineering projects. As depicted in Table 3, different projects have vastly different numbers of manually created trace links. For example, projects such as Em and xL have a very limited set of trace links with which to train MT-BERT.

6.3 RQ5: Evaluation of MT-BERT

We start by evaluating the efficacy of MT-BERT by refining RQ5 "Does MT-BERT increase the accuracy of generated trace links in comparison to multi-lingual IR models?" into two sub-questions. The first evaluates the use of different language models within MT-BERT, while the second compares the best MT-BERT result against the previously reported best Information Retrieval model. The research questions are structured as follows:

- **RQ5.1** How does the choice of Language Model impact the accuracy of trace links generated using MT-BERT?
- **RQ5.2** How does the accuracy of MT-BERT compare against WEG_{tr}?

To provide adequate training examples for MT-BERT, we merged all 14 Chinese projects from Table 3b into a single compound dataset. We then split this compound dataset into ten equal buckets, and utilized eight folds for the training dataset, one fold for validation, and one for testing. This resulted in a training set of 3,683 links, a validation set of 461 links, and a testing set of 462 links. We used this compound dataset to emulate the scenario in which MT-BERT is trained with a relatively large dataset. As this may not always be the case, we investigate resource limited scenarios in RQ6. To compare MT-BERT against the IR models, we selected WEG_{tr} for the baseline as it outperformed CLG and VSM_{tr} in most cases. To make a fair comparison between MT-BERT and WEG_{tr}, we used exactly the same multi-project data for training, validating, and testing WEG_{tr}. To conduct MT-BERT model training, we used 200 training epochs for all of the following experiments. We evaluated the model with validation dataset during the training, and then selected the best model based on the validation results. This model was then applied against the test set. We used a learning rate of 1E-5 and a batch size of 16. We run each of the experiments three times with distinct data splits and report the averaged results.

6.3.1 RQ5.1: The Choice of Language Model

As depicted in Fig. 5, MT-BERT leverages a multilingual LM to encode the artifact text. Training an LM from scratch is expensive, because it requires not only a massive text corpus but also very large computation resources. This is not practical for traceability purposes across diverse projects. Therefore, we reused the LM provided by HuggingFace produced by large resource-rich companies. To investigate the use of different LMs we conducted an experiment using three different models, including a knowledge distilled multilingual LM (Sanh et al. 2019) (LM-1), a multilingual LM without knowledge distillation (LM-2), and a monolingual LM (LM-3). LM-1 and LM-2 are published by HuggingFace and Google respectively. We chose these two models because they are the largest uncased multilingual LMs available and constructed from 104 languages. The third language model (LM-3), released by Google, was chosen as it is the most widely used monolingual LM.

Results achieved from generating trace links against the test data are reported in Table 7. They show that MT-BERT used with the knowledge distilled multilingual language model

Table 7 Trace link generation accuracy is compared for three different language models (LM), serving as encoders of MT-BERT, against the previously reported best multilingual IR model

Approach	Language Model	F1	AP
MT-BERT (LM-1)	Knowledge distilled multilingual BERT LM	0.366	0.331
MT-BERT (LM-2)	Non-Distilled multilingual BERT LM	0.322	0.311
MT-BERT (LM-3)	Monolingual BERT LM	0.234	0.237
WEG_{tr}	n/a	0.169	0.085

(LM-1) achieved the best performance, and outperformed the other multilingual model (LM-2) by 4.0% F1 score and 3.3% AP. Both multilingual language models outperformed the monolingual one, with the top-performing multilingual model (LM-1) outperforming the monolingual model (LM-3) by 5.9% F1 and 7.3% AP. Given these results we adopted the knowledge distilled language model (LM-1) for use in our final MT-BERT architecture.

6.3.2 RQ5.2: Comparison of MT-BERT against WEG$_{tr}$

Our results in Table 7 also show that all three MT-BERT models outperformed the WEG$_{tr}$ approach. The knowledge distilled language model (LM-1) returned non-trivial improvements of 23.9% F1 and 27.8% AP scores over the IR-based WEG$_{tr}$ approach. We therefore address RQ5.2 by concluding that MT-BERT can achieve better performance than WEG$_{tr}$ when provided with a relatively large set of training examples. Trace link accuracy returned by WEG$_{tr}$ was relatively low, most likely because of the size and complexity of the artifacts in the compound dataset. These results indicate that WEG$_{tr}$ does not scale well for tracing in large projects.

6.3.3 RQ6: MT-BERT in Resource Limit Scenarios

Deep learning approaches rely upon big data; however, many software projects have insufficient sets of trace links for training a dedicated MT-BERT. For example, the Em, QMI and xL projects each have very few links, making it practically impossible to train a deep learning model using the data in those projects alone. To tackle this issue, we trained MT-BERT on training data from all other projects sharing the same languages (i.e., English-Chinese in our data set), and then used the trained model to generate links for each individual project. We split each individual project into training, validation, and test datasets. For projects with sufficient links, we applied a normal 8/1/1 split; however, for projects containing fewer than 60 links, we split the dataset into equally sized training, validation, and testing sets to ensure that we had enough test examples to conduct our evaluation. To train the model and select the best configuration we took the training and validation datasets from all from 14 Chinese projects and constructed MT-BERT using the knowledge distilled LM (LM-1). We then used this model to evaluate the accuracy of links for each project's testing set. To evaluate whether the knowledge from adjacent projects actually improved the tracing performance, we also trained an individual model for each project using only the training data from its own project dataset. This set of project-specific models served as our control group. All other configurations were identical for both treatments.

We report results from this experiment in Table 8. This table orders projects according to increasing project size. In contrast to Table 7 which reports results for all links and weights each link equally, this table reports results for each individual project.

Table 8 M-TBERT trained using training data for project artifacts of 14 Chinese projects with performance evaluated on the test set for each project independently

	Multi-Project General MT-BERT(All)		Project Dedicated MT-BERT(Project)		WEG_{tr}		No. of Links
	F1	AP	F1	AP	F1	AP	
Em	0.322	0.284	0.299	0.221	0.507	0.460	32
QMI	0.338	0.259	0.183	0.102	0.681	0.567	35
xL	0.483	0.338	0.170	0.099	0.583	0.597	52
QMA	0.415	0.408	0.346	0.277	0.568	0.595	71
NC	0.680	0.628	0.529	0.517	0.620	0.608	99
We	0.228	0.165	0.224	0.124	0.446	0.389	159
Pe	0.811	0.872	0.953	0.988	0.768	0.748	160
Na	0.305	0.254	0.331	0.271	0.396	0.377	161
AR	0.532	0.567	0.327	0.242	0.538	0.489	167
Ca	0.253	0.096	0.270	0.193	0.381	0.351	273
Sa	0.626	0.515	0.504	0.425	0.637	0.697	275
Rax	0.499	0.355	0.405	0.355	0.362	0.283	571
Dr	0.282	0.197	0.237	0.163	0.216	0.140	1161
BK	0.423	0.329	0.373	0.507	0.190	0.085	1179

We first compare the use of the multi-project LM (MT-BERT-All) versus the project-specific LM (MT-BERT-Project) and observe that MT-BERT-All achieved better performance in 10 out of 14 projects returning an average F1 score of approximately 0.443 and AP score of 0.376 versus 0.368 and 0.320 for the project-dedicated training. This indicates that training the MT-BERT using data from other projects was very beneficial. The p-value and effect size in Table 9 indicates that MT-BERT-All is significantly better than MT-BERT-Project (p-value 0.021¡0.05) with a 'median' effect size ($0.4 > 0.3$). Intuitively, the larger the training data, the more likely it would be for project-dedicated MT-BERT (MT-BERT-Project) to perform well. We evaluated this intuition using the Pearson correlation coefficient which returns a value between -1 and 1, where -1 indicates a perfect negative linear correlation, 0 indicates no linear correlation, and 1 indicates perfect positive linear correlation. Testing for correlations between accuracy and number of links, using the MT-BERT-Project returned negligible correlations (F1:-0.060, AP:0.083) indicating that project size, represented by the number of links, was not a dominant differentiating factor.

Next, we compared MT-BERT-All against the WEG_{tr} results per project. These results show that WEG_{tr} outperformed MT-BERT-All in 11 out of 14 projects, with a more pronounced improvement in the smaller projects. The P-value and effect size in Table 9 shows that MT-BERT-All is not statistically better than WEG_{tr} with a p-value of 0.245 and effective size -0.289. However, in projects with larger numbers of links, WEG_{tr} underperformed. As a result, the Pearson Correlation returned a moderate negative correlation between the number of links in a project and WEG_{tr} accuracy (F1: -0.166, AP:-0.227).

Taking this information together confirms our earlier findings that WEG_{tr} performs quite well on smaller datasets but does not scale up well for larger projects.

Furthermore, for the datasets in our project, the use of MT-BERT trained on all available datasets generally improves trace accuracy for larger sized projects. For example, with

Table 9 P-values of wilcoxon signed-rank test and their effect size for IR models

	MT-BERT(Project)		WEG^*	
	P-Value	Effect Size	P-Value	Effect Size
MT-BERT(All)	0.021	0.400	0.245	-0.289

our datasets, we achieved the highest overall accuracy when applying MT-BERT-All to the largest three projects and WEG_{tr} to the others (i.e., F1: 0.523, AP: 0.482). As industrial projects can have hundreds of thousands of trace links, MT-BERT is highly likely to outperform WEG_{tr} in large industrial projects.

6.4 Extension to Other Languages

We also conducted experiments with the three non-Chinese projects. We applied two different treatments. In the first case, we trained an MT-BERT model using all 14 Chinese projects, and used this model to generate trace links for each of the three non-Chinese projects. In the second case, we trained MT-BERT using the limited data available from each project. As each project had approximately 30 links, we provided MT-BERT ten links for training and used the remaining ones for testing.

The results are shown in Table 10, and indicate that the MT-BERT model (column 1) trained on the Chinese projects resulted in improved accuracy over the project-specific MT-BERT (column 2) for all the Korean, Japanese and German projects, with improvements in AP score of 0.121, 0.088 and 0.036 respectively. As we can see, the German project benefited less than the other projects.

One explanation is that as Korean and Japanese are character based language and share some characters with Chinese, these projects are more likely to benefit from reusing the Chinese pretrained MT-BERT model than the German project.

Finally, we find WEG_{tr} (column 3) achieved significantly better results than both cases that used MT-BERT on these projects. This further supports our previous conclusion that multilingual IR models outperform MT-BERT models for small projects.

7 Threats to Validity

There are several threats to validity in this study. First, we used Google as our black-box translator. As the vocabulary selection strategy of the NMT has a direct impact on the final

Table 10 Average Precision achieved when applying MT-BERT on the three non-Chinese projects of Korean, Japanese and German projects

	General MT-BERT Trained with Chinese Project Data	Project dedicated MT-BERT	WEG_{tr}
Ko (Korean)	0.348	0.227	0.510
Ci (Japanese)	0.273	0.185	0.479
Ab (German)	0.456	0.420	0.695

The first model trained MT-BERT on 14 Chinese projects, the second trained it using project-specific training data. WEG_{tr} outperformed the MT-BERT model in all cases, likely due to the projects' small size

trace link quality, results could be different if other types of machine translation methods are applied. However, we chose Google translation as it has been empirically shown to deliver high quality translations across numerous languages. Another important threat is that the training material used for CLG was composed of general documents from Wikipedia and did not include domain specific corpora. We made this decision in order to deliver a more generalized solution, and because collecting a domain specific corpus for all 17 projects would have been prohibitively expensive. CLG might perform better than WEG if a domain-specific corpus of technical documents had been available and used for training purposes. An external threat was introduced by limiting the raw artifacts to the coverage area of intermingled links to alleviate the link sparsity issue (see Table 3b). This enabled us to focus on the IBA-impacted traces, but reduced the number of participating artifacts, thereby potentially inflating AP scores for all results. Also, we did not yet explore the impact of embedding size on CLG. On the other hand, our experiments showed that WEG_{tr} still outperformed CLG, when equal sized embeddings were used. The reality is that larger monolingual embeddings are more readily available, and should be pragmatically leveraged. We leave experimentation with different sized embeddings to future work. Also, as reported in Tables 10 we explored the efficacy of training MT-BERT on individual projects and reported that training on a broader set of projects was more effective. However, some industrial projects are much larger than even the largest OSS dataset used in our work, and project-specific training might be very effective in those cases. Similarly, all of the projects based on non-Chinese languages were quite small, and results might differ if larger projects had been available. While much larger multilingual projects do exist as OSS, we did not find any with sufficient issues and commits to support our study.

As a result, while we have explored the question of traceability in Chinese-English projects, further work is needed to determine if these results can be generalized to other languages.

8 Related Work

We have extensively described a wide body of work related to the general problem of cross-lingual translation throughout the paper. This section therefore focuses on the more limited limited body of work concerning the use of multiple languages in the software development environment. This phenomenon tends to occur in two primary settings – first in organizations in which English is not the primary language of many stakeholders, but is the language of choice for supporting the development process; and second, in global software development environments with geographically dispersed teams speaking multiple languages. As Abufardeh and Magel (2010) points out, this kind of geographical localization for development teams is a critical element of the success of multi-national projects.

Multilingual problems in global software development (GSD) have been identified and extensively discussed. Although English is widely accepted as an official languages in most international IT corporations, the practice of utilizing a second language is quite common. For example, Lutz (2009) investigated the issue in Siemens' English-German work space and pointed out that utilizing English as the primary language for non-native speakers can lead to misunderstandings in both oral and written communication. Treude et al. investigated the problem of software analytics in multilingual projects containing a mixture of Portuguese and English. They found that the use of English terms in Portuguese artifacts reduced the accuracy of Portuguese pos-taggers and further impacted the analytic tasks relying on these tools (Treude et al. 2015).

Researchers have proposed different methods to address the multi-lingual linguistic problem. One branch of studies has focused on developing best-practices (Krishna et al. 2004) to enhance the work quality and efficiency. While others have proposed using machine translation as a solution for minimizing the misunderstandings introduced as a byproduct of the multi-lingual environment (Calefato et al. 2011; Moulin et al. 2009; Cleland-Huang et al. 2011). Wouters et al. highlight a collaborative software development challenge for the European Aeronautic Defense and Space Company (Wouters et al. 2013), where domain experts in different fields from multiple countries needed to interpret domain terminologies during the collaboration. They proposed an artifact construction system, which leveraged a machine translation service to maintain a domain ontology. When a domain expert creates new artifacts, their output will be automatically replicated as artifacts in other languages. Similarly, Calefato et al. discussed the adoption of machine translation during the distributed software requirements engineering process (Calefato et al. 2010; Calefato et al. 2011), where engineers are from different locations and speak different languages. These studies focused on addressing the same semantic gap introduced by the use of a multilingual vocabulary that we have addressed in our study. Whereas they proposed the use of machine translation during the artifacts construction stage in order to avoid creating intermingled bilingual artifacts in the first place, our approach processes the intermingled bilingual artifacts directly in order to achieve accurate traceability.

Cruz et al. explored the impact of vague words in multilingual requirements and addressed this issue by applying a black list (Cruz et al. 2017). The explored ways to migrate the vagueness detection method from English to Portuguese and Spanish. They compared two approaches for overcoming the language gap – first, by translating the blacklist of words, and second, by translating the whole artifact written in a foreign language back into English. They reported that translating the entire artifact achieved better results, as the translator was able to take into consideration the context of vague words and create a higher quality translation. This result highlights the importance of context in multilingual artifact analysis. Xia et al. (2014) proposed addressing the translation ambiguity problem by building a voting mechanism based on multiple translators. Our study goes further, by discussing the pitfalls of machine translation within different contexts, and proposing a deep learning model to handle the context directly without the need for machine translation.

Other researchers, such as Monti et al. (2013) and Hilgert et al. (2014) have built cross-lingual ontologies and domain specific cross-lingual dictionaries to mitigate the semantic gaps introduced by multilingual vocabularies. Hilgert et al., proposed a method to automatically extract bilingual vocabulary from parallel corpora, and then transforming the intermingle bilingual artifacts into monolingual artifacts by replacing the concepts written in the minor language with those from the primary language. Xu et al. (2018) adopted a similar technique for tackling the cross-lingual domain retrieval problem for questions on stackoverflow, by building a domain-specific cross-lingual vocabulary to support the online translation of Chinese engineering questions. We have similarly tackled the problem of multilingual language use in software artifacts by applying diverse machine-learning solutions that are designed to compensate for the use of multiple languages to achieve more accurate traceability.

9 Conclusion

The work in this paper was initially motivated by the needs of our industrial collaborators who were seeking enterprise-wide traceability solutions across software repositories

containing artifacts written in a combination of English and Chinese. We have systematically evaluated the use of basic and semantically-imbued information retrieval techniques, as well as deep learning techniques based on pre-trained language models.

We first explored the use of intermingled Chinese and English terms across 14 different projects and identified common usage patterns. We then showed that using a preprocessing translation step in IBA projects in conjunction with three commonly used trace models improved accuracy of the generated trace links thereby highlighting the importance of addressing the multilingual problem in automated trace link creation algorithms. We then explored two different approaches for further addressing the semantic gap caused by multilingual terminology. We first proposed three GVSM based methods used in conjunction with word embedding to address the IBA vocabulary issue. Our experiments showed that, WEG^{*}_{tr} performed best but that CLG and WEG were potentially cost-effective alternatives because it is easier, more effective, and less costly to train a word embedding based model than an NMT translator. Furthermore, an internally trained CLG and WEG model could potentially include domain-specific terminology, thereby potentially boosting its performance.

We then proposed and evaluated MT-BERT, a tracing model based on a multilingual Language Model. We found that MT-BERT outperformed IR models in large scale projects, making it particularly well suited to many industrial environments. Finally, we showed the MT-BERT's performance could be improved when additional training examples were provide from adjacent projects which are intermingled with the same language as the target project.

Given the results reported in this paper, the MT-BERT approach offers significant promise for industrial organizations who need to perform traceability across multilingual artifacts especially when large numbers of trace links are available for training purposes.

Acknowledgements The work described in this paper has been partially funded by United States National Science Foundation grants CCF-1649448 and SHF-1901059.

References

EF EPI (2019) EF English Proficiency Index

Fasttext (2021) Word vectors for 157 languages · fasttext

Double Blinded (2020) All information is blinded due to current submission under double blind review. the paper is available upon request to the associate editors of the msr emse special edition

Abufardeh S, Magel K (2010) The impact of global software cultural and linguistic aspects on global software development process (gsd): Issues and challenges. In: 4th International conference on new trends in information science and service science. pp 133–138

Ali N, Guéhéneuc Y, Antoniol G (2013) Trustrace: Mining software repositories to improve the accuracy of requirement traceability links. IEEE Trans Softw Eng 39(5):725–741

Almasri M, Berrut C, Chevallet J (2016) A comparison of deep learning based query expansion with pseudo-relevance feedback and mutual information. In: Advances in information retrieval - 38th European conference on IR research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings. pp 709–715

Antoniol G, Canfora G, Casazza G, Lucia AD, Merlo E (2002) Recovering traceability links between code and documentation. IEEE Trans Software Eng 28(10):970–983

Asuncion HU, Asuncion A, Taylor RN (2010) Software traceability with topic modeling. In: 32nd ACM/IEEE International conference on software engineering (ICSE). pp 95–104

Asuncion HU, Taylor RN (2012) Automated techniques for capturing custom traceability links across heterogeneous artifacts. In: Software and systems traceability. pp 129–146

Bird S (2006) NLTK: the natural language toolkit. In: ACL 2006, 21st International conference on computational linguistics and 44th annual meeting of the association for computational linguistics, Proceedings of the Conference, Sydney, Australia, 17-21 July 2006

Calefato F, Lanubile F, P Minervini and (2010) Can real-time machine translation overcome language barriers in distributed requirements engineering? In: 2010 5th IEEE International conference on global software engineering. IEEE, pp 257–264

Calefato F, Lanubile F, Prikladnicki R (2011) A controlled experiment on the effects of machine translation in multilingual requirements meetings. In: 6th IEEE International conference on global software engineering, ICGSE 2011, Helsinki, Finland, August 15-18, 2011. pp 94–102

Cleland-Huang J, Czauderna A, Dekhtyar A, Gotel O, Hayes JH, Keenan E, Leach G, Maletic JI, Poshyvanyk D, Shin Y, Zisman A, Antoniol G, Berenbach B, Egyed A, Måder P (2011) Grand challenges, benchmarks, and tracelab: developing infrastructure for the software traceability research community. In: TEFSE'11, Proceedings of the 6th International workshop on traceability in emerging forms of software engineering, May 23, 2011, Waikiki, Honolulu, HI, USA. pp 17–23

Cleland-Huang J, Gotel O, Hayes JH, Mäder P, Zisman A (2014) Software traceability: trends and future directions. In: FOSE. pp 55–69

Cleland-Huang J, Rahimi M, Måder P (2014) Achieving lightweight trustworthy traceability. In: Proceedings of the 22nd ACM SIGSOFT International symposium on foundations of software engineering, (FSE-22), Hong Kong, China, November 16 - 22, 2014. pp 849–852

Conneau A, Lample G, Ranzato M, Denoyer L, Jégou H. (2017) Word translation without parallel data. arXiv:1710.04087

Conneau A, Lample G, Rinott R, Williams A, Bowman SR, Schwenk H, Stoyanov V (2018) Xnli: Evaluating cross-lingual sentence representations. arXiv:1809.05053

Cover TM, Thomas JA (2006) Elements of information theory (Wiley series in telecommunications and signal processing). Wiley-Interscience, New York

Cruz BD, Jayaraman B, Dwarakanath A, McMillan C (2017) Detecting vague words & phrases in requirements documents in a multilingual environment. In: 2017 IEEE 25th International requirements engineering conference (RE). pp 233–242. IEEE

Devlin J, Chang M-W, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding arXiv:1810.04805

Dhingra B, Zhou Z, Fitzpatrick D, Muehl M, Cohen WW (2016) Tweet2vec: Character-based distributed representations for social media. In: Proceedings of the 54th annual meeting of the association for computational linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers

Fu Y (2021) Who offers the best chinese-english machine translation? a comparison of google, microsoft bing, baidu, tencent, sogou, and netease youdao

Google-Research (2019) Github Repository: Multilingual Models google-research/bert

Gotel O, Cleland-Huang J, Huffman Hayes J, Zisman A, Egyed A, Grünbacher P., Antoniol G (2012) The quest for ubiquity: A roadmap for software and systems traceability research. In: 21st IEEE International requirements engineering conference (RE). pp 71–80

Gotel OCZ, Finkelstein A (1994) An analysis of the requirements traceability problem. In: Proceedings of the first IEEE international conference on requirements engineering, ICRE '94, Colorado Springs, Colorado, USA, April 18-21, 1994. pp 94–101

Gouws S, Bengio Y, Corrado G (2015) Bilbowa: Fast bilingual distributed representations without word alignments. In: Proceedings of the 32nd International conference on machine learning, ICML 2015, Lille, France, 6-11 July 2015. pp 748–756

Guo J, Cheng J, Cleland-Huang J (2017) Semantically enhanced software traceability using deep learning techniques. In: Proceedings of the 39th international conference on software engineering, ICSE 2017, Buenos Aires, Argentina, May 20-28, 2017. pp 3–14

Guo J, Cleland-Huang J, Berenbach B (2013) Foundations for an expert system in domain-specific traceability. In: 21st IEEE International requirements engineering conference, RE 2013, Rio de Janeiro-RJ, Brazil, July 15-19, 2013. IEEE Computer Society, pp 42–5

Harris Z (1954) Distributional structure. Word 10(23):146–162

Hayes JH, Dekhtyar A, Sundaram SK (2006) Advancing candidate link generation for requirements tracing: The study of methods. IEEE Trans Software Eng 32(1):4–19

Hilgert L, Lopes L, Freitas A, Vieira R, Hogetop D, Vanim A (2014) Building domain specific bilingual dictionaries. In: Proceedings of the ninth international conference on language resources and evaluation (LREC'14), 2014, Islândia

Hofmann T (1999) Probabilistic latent semantic indexing. In: SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval, August 15-19, 1999, Berkeley, CA, USA. pp 50–57

Jenkins J (1999) New ideographs in unicode 3.0 and beyond. In: Proceedings of the 15th international unicode conference C, vol 15. pp 1–2

Johnson M, Schuster M, Le QV, Krikun M, Wu Y, Chen Z, Thorat N, Viégas F, Wattenberg M, Corrado G et al (2017) Google's multilingual neural machine translation system: Enabling zero-shot translation. Trans Assoc Comput Linguist 5:339–351

Jones E, Oliphant T, Peterson P et al (2001) SciPy: Open source scientific tools for Python. [Online; accessed <today>]

Joulin A, Bojanowski P, Mikolov T, Jégou H., Grave E (2018) Loss in translation: Learning bilingual word mapping with a retrieval criterion. In: Proceedings of the 2018 conference on empirical methods in natural language processing, Brussels, Belgium, October 31 - November 4, 2018. pp 2979–2984

Kailath T (1967) The divergence and bhattacharyya distance measures in signal selection. IEEE Trans Commun Technol 15(1):52–60

Khandkar SH (2009) Open coding. University of Calgary, 23:2009

Krishna S, Sahay S, Walsham G (2004) Managing cross-cultural issues in global software outsourcing. Commun ACM 47(4):62–66

Liu Y, Lin J, Cleland-Huang J (2020) Traceability support for multi-lingual software projects. In: Kim S, Gousios G, Nadi S, Hejderup J (eds) MSR '20: 17th International conference on mining software repositories, Seoul, Republic of Korea, 29-30 June, 2020. ACM, pp 443–454

Liu Y, Lin J, Zeng Q, Jiang M, Cleland-Huang J (2020) Towards semantically guided traceability. In: 2020 IEEE 28th International requirements engineering conference (RE). pp 328–333. IEEE

Lohar S, Amornborvornwong S, Zisman A, Cleland-Huang J (2013) Improving trace accuracy through data-driven configuration and composition of tracing features. In: Joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering, ESEC/FSE'13, Saint Petersburg, Russian Federation, August 18-26, 2013. pp 378–388

Lormans M, Van Deursen A (2006) Can lsi help reconstructing requirements traceability in design and test? In: Conference on software maintenance and reengineering (CSMR'06). IEEE, pp 10–pp

Lucia AD, Fasano F, Oliveto R, Tortora G (2007) Recovering traceability links in software artifact management systems using information retrieval methods. ACM Trans Softw Eng Methodol 16(4)

Lutz B (2009) Linguistic challenges in global software development: Lessons learned in an international SW development division. In: 4th IEEE International conference on global software engineering, ICGSE 2009, Limerick, Ireland, 13-16 July, 2009. pp 249–253

Mäder P, Gotel O (2012) Towards automated traceability maintenance. J Syst Softw 85(10):2205–2227

Meeker M, Wu L (2018) Internet trends 2018

Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv:1301.3781

Mikolov T, Grave E, Bojanowski P, Puhrsch C, Joulin A (2018) Advances in pre-training distributed word representations. In: Proceedings of the International conference on language resources and evaluation (LREC 2018)

Monti J, Monteleone M, Di Buono MP, Marano F (2013) Natural language processing and big data-an ontology-based approach for cross-lingual information retrieval. In: 2013 International conference on social computing. IEEE, pp 725–731

Moulin C, Sugawara K, Fujita S, Wouters L, Manabe Y (2009) Multilingual collaborative design support system. In: Proceedings of the 13th International conference on computers supported cooperative work in design, CSCWD 2009, April 22-24, 2009, Santiago, Chile. pp 312–318

Muhr M, Kern R, Zechner M, Granitzer M (2010) External and intrinsic plagiarism detection using a cross-lingual retrieval and segmentation system. In: Notebook papers of CLEF 2010 LABs and workshops

Oliveto R, Gethers M, Poshyvanyk D, Lucia AD (2010) On the equivalence of information retrieval methods for automated traceability link recovery. In: The 18th IEEE International conference on program comprehension, ICPC 2010, Braga, Minho, Portugal, June 30-July 2, 2010. pp 68–71

Pawelka T, Juergens E (2015) Is this code written in english? a study of the natural language of comments and identifiers in practice. In: 2015 IEEE International conference on software maintenance and evolution (ICSME). IEEE, pp 401–410

Rath M, Rendall J, Guo JLC, Cleland-Huang J, Mäder P (2018) Traceability in the wild: automatically augmenting incomplete trace links. In: Proceedings of the 40th international conference on software engineering, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018. pp 834–845

Rempel P, Mäder P, Kuschke T, Cleland-Huang J (2015) Traceability gap analysis for assessing the conformance of software traceability to relevant guidelines. In: Software engineering & management 2015, Multikonferenz der GI-Fachbereiche Softwaretechnik (SWT) und Wirtschaftsinformatik, Dresden, Germany. pp 120–121

Ruder S, Vuli'c I, Sogaard A (2017) A survey of cross-lingual word embedding models

Sanh V, Debut L, Chaumond J, Wolf T (2019) Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv:1910.01108

Shin Y, Hayes JH, Cleland-Huang J (2015) Guidelines for benchmarking automated software traceability techniques. In: 8th IEEE/ACM International symposium on software and systems traceability, SST 2015, Florence, Italy, May 17, 2015. pp 61–67

Spanoudakis G, Zisman A, Pérez-Miñana E, Krause P (2004) Rule-based generation of requirements traceability relations. J Syst Softw 72(2):105–127

Tang G, Xia Y, Zhang M, Li H, Zheng F (2011) CLGVSM: adapting generalized vector space model to cross-lingual document clustering. In: Fifth International joint conference on natural language processing, IJCNLP 2011, Chiang Mai, Thailand, November 8-13, 2011. pp 580–588

Trec-Kba, trec-kba/many-stop-words (2021)

Treude C, Prolo CA, Figueira Filho F (2015) Challenges in analyzing software documentation in portuguese. In: 2015 29th Brazilian symposium on software engineering. IEEE, pp 179–184

Tsatsaronis G, Panagiotopoulou V (2009) A generalized vector space model for text retrieval based on semantic relatedness. In: EACL 2009, 12th conference of the european chapter of the association for computational linguistics, Proceedings of the Conference, Athens, Greece, March 30 - April 3, 2009. pp 70–78

Tsatsaronis G, Varlamis I, Vazirgiannis M (2010) Text relatedness based on a word thesaurus. J Artif Intell Res 37:1–39

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. In: Advances in neural information processing systems. pp 5998–6008

Vulic I (2017) Cross-lingual syntactically informed distributed word representations. In: Proceedings of the 15th conference of the european chapter of the association for computational linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers. pp 408–414

Vulic I, Moens M (2015) Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In: Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, Santiago, Chile, August 9-13, 2015. pp 363–372

Wada T, Iwata T (2018) Unsupervised cross-lingual word embedding by multilingual neural language models. arXiv:1809.02306

Wong SKM, Ziarko W, Raghavan VV, Wong PCN (1989) Extended boolean query processing in the generalized vector space model. Inf Syst 14(1):47–63

Wong SKM, Ziarko W, Wong PCN (1985) Generalized vector space model in information retrieval. In: Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval, Montréal, Québec, Canada, June 5-7, 1985. pp 18–25

Woolson R (2007) Wilcoxon signed-rank test. Wiley encyclopedia of clinical trials. pp 1–3

Wouters L, Kaeri Y, Sugawara K (2013) Multi-domain multi-lingual collaborative design. In: Proceedings of the 2013 IEEE 17th International conference on computer supported cooperative work in design (CSCWD), IEEE, pp 269–274

Wu Y, Schuster M, Chen Z, Le QV, Norouzi M, Macherey W, Krikun M, Cao Y, Gao Q, Macherey K, Klingner J, Shah A, Johnson M, Liu X, Kaiser L, Gouws S, Kato Y, Kudo T, Kazawa H, Stevens K, Kurian G, Patil N, Wang W, Young C, Smith J, Riesa J, Rudnick A, Vinyals O, Corrado G, Hughes M, Dean J (2016) Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv:1609.08144

Xia X, Lo D, Wang X, Zhang C, Wang X (2014) Cross-language bug localization. In: Proceedings of the 22nd International conference on program comprehension. pp 275–278

Xu B, Xing Z, Xia X, Lo D, Li S (2018) Domain-specific cross-language relevant question retrieval. Empir Softw Eng 23(2):1084–1122

Ye X, Qi Z, Massey D (2015) Learning relevance from click data via neural network based similarity models. In: 2015 IEEE International conference on big data, Big Data 2015, Santa Clara, CA. pp 801–806

Zhao T, Cao Q, Sun Q (2017) An improved approach to traceability recovery based on word embeddings. In: 24th Asia-pacific software engineering conference, APSEC 2017, Nanjing, China, December 4-8, 2017. pp 81–89