УДК 004.855.5

ПОСТРОЕНИЕ МОДЕЛИ МАШИННОГО ОБУЧЕНИЯ ДЛЯ ЗАДАЧИ КЛАССИФИКАЦИИ СТЕПЕНИ КРИТИЧНОСТИ CVE-УЯЗВИМОСТЕЙ

А. К. Доронин
магистр технических наук, аспирант
Белорусский государственный университет информатики
и радиоэлектроники
В. А. Липницкий
доктор технических наук, профессор
Военная академия Республики Беларусь

В статье рассматривается использование методов машинного обучения в сочетании с алгоритмом представления слов в многомерном векторном пространстве GloVe для задачи предсказания критичности уязвимости, основываясь лишь на ее текстовом описании. В качестве набора данных для анализа и обучения используется база данных об уязвимостях NVD. В статье приведен анализ записей об уязвимостях, механизм оценки уязвимостей, на основании чего обосновывается выбор признаков для обучения модели. Описываются различные подходы и методы для построения векторных представлений слов, обосновывается выбор векторов, построенных алгоритмом GloVe. Также приводится сравнительный анализ работы модели на наборах векторов GloVe различных размерностей, сделаны выводы об использовании 50-размерных векторов GloVe. Приводятся несколько возможных дальнейших практических применений обученной модели.

Ключевые слова: машинное обучение, конволюционные нейронные сети, анализ данных, уязвимости компьютерных систем, оценка степени критичности уязвимости, векторное представление слов, обработка текста.

Введение

Каждый день в мире становится известно около 20 новых кибер-уязвимостей. Каждый день в IT-сфере появляются сведения примерно о 20 новых киберуязвимостях. Практически все они связаны с различными недостатками реализации программного обеспечения. Злоумышленники могут использовать эти уязвимости для запуска атаки, инициации системного сбоя, организации доступа к конфиденциальной информации или получения удаленного доступа к системе. Некоторые уязвимости несут в себе большой потенциальный ущерб, в то время как другие имеют малый интерес у злоумышленников или же интерес отсутствует вовсе. Для офицера информационной безопасности может оказаться сложным быстро и верно оценить, какие уязвимости имеют более высокий приоритет для исправления, а какие – менее высокий. Ландшафт киберугроз быстро меняется, и для многих организаций жизненно важно постоянно обновлять ПО и активно работать над улучшением безопасности.

Многие уязвимости относятся к так называемым уязвимостям нулевого дня (0-day), что означает: данная уязвимость была обнаружена до того, как компания-разработчик ПО узнала о ее существовании. 90% эксплойтов (программ, позволяющих использовать уязвимость) обычно доступны в течение недели с момента раскрытия уязвимости, подавляющее большинство из них – в течение нескольких дней. Таким образом, автоматическая ранняя оценка критичности уязвимости может помочь офи-

церам безопасности заранее обнаружить возможные угрозы и предпринять меры по противодействию.

Американская Национальная база данных уязвимостей (NVD) как набор данных для анализа

Американская Национальная база данных уязвимостей NVD (National Vulnerabilities Database) основана на списке уязвимостей из проекта CVE, запущенном MITRE в 1999 г. [1].

CVE (Common Vulnerabilities and Exposures) – это общедоступный список записей, каждый из которых содержит идентификационный номер, описание и, по крайней мере, одну общедоступную ссылку для каждой уязвимости. Записи из списка CVE используются в многочисленных продуктах и услугах по кибербезопасности со всего мира, в том числе и NVD [2].

Все записи в базе данных NVD имеют 14 полей, из них основными являются следующие:

- CVE Id: уникальный ID (номер) уязвимости (например, CVE-2011-1585)
- Date published: дата первой публикации о данной уязвимости
- Date modified: последняя дата изменения записи
- Summary: текстовое описание уязвимости
- Оставшиеся поля относятся к параметрам оценки уязвимости, выставляемой в соответствии со стандартом CVSS [2]:
- CVSS Base: базовая оценка уязвимости (десятичное значение от 0 до 10, например, 9.5)
- CVSS Impact: оценка элементов воздействия (десятичное значение от 0 до 10, например, 9.5)
- CVSS Exploit: оценка возможности эксплуатации (десятичное значение от 0 до 10, например, 9.5)
- CVSS Access vector (AV): вектор доступа (локальный/сетевой/локально-сетевой)
- CVSS Access complexity (AC): сложность доступа (низкая/средняя/высокая)
- CVSS Authentication (Au): уровень требуемой аутентификации (нулевая/однократная/многократная)
- CVSS Confidentiality impact (C): воздействие на конфиденциальность (полное/частичное/нулевое)
- CVSS Integrity impact (I): воздействие на целостность (полное/частичное/нулевое)
- CVSS Availability impact (A): воздействие на доступность (полное/частичное/нулевое)
- CVSS Vector: базовый вектор уязвимости содержит в себе значения других полей (например, AV:N/AC:M/Au:N/C:N/I:P/A:C)

В таблице 1 отражен пример записи об уязвимости CVE-2008-4250, известной также как MS-08-067 [3].

Таблица 1 – Пример записи в БД NVD

Название поля	Значение
CVE ID	CVE-2008-4250
Date published	2008-10-23
Date modified	2018-10-12
CVSS Base	10.0
CVSS Impact	10.0
CVSS Exploit:	10.0

Окончание таблицы 1

Название поля	Значение
CVSS Access vector	Network
CVSS Access Complexity	Low
CVSS Authentication	None
CVSS Score	10
CVSS Confidentiality Impact	Complete
CVSS Integrity Impact	Complete
CVSS Availability Impact	Complete
CVSS Vector	AV:N/AC:L/Au:N/C:C/I:C/A:C
Summary	The Server service in Microsoft Windows 2000 SP4, XP SP2 and SP3, Server 2003 SP1 and SP2, Vista Gold and SP1, Server 2008, and 7 Pre-Beta allows remote attackers to execute arbitrary code via a crafted RPC request that triggers the overflow during path canonicalization, as exploited in the wild by Gimmiv.A in October 2008, aka "Server Service Vulnerability."

Текстовое описание в качестве источника обучения модели машинного обучения

С помощью алгоритмов машинного обучения можно построить модель, бинарно классифицирующую уязвимости как критичные или некритичные по тем или иным признакам.

В исследовании [4] компании Symantec показано, что наиболее информативным источником признаков для последующего анализа и применения методов машинного обучения является текстовое описание уязвимости в БД NVD.

Описание является аккумулятором информации из других признаков. При этом оно обычно составлено кратко и содержит в себе описание ключевых особенностей уязвимости.

В данной работе мы строим модель машинного обучения с целью определения степени критичности уязвимости, также используя в качестве источника обучения ее текстовое описание из БД NVD.

Степень критичности как целевой признак для модели обучения

Степень критичности уязвимости в NVD выражается в трех признаках:

cvss_impact (воздействие) – оценка воздействия на компонент в случае ее эксплуатации. Оценивается худший результат, который наиболее прямо и предсказуемо связан с атакой, использующей уязвимость.

cvss_exploit (эксплуатируемость) – оценка возможности эксплуатации уязвимости, отражающая суммарные показатели свойств уязвимости, которые приводят к успешной атаке.

cvss_base- средняя оценка *cvss_impact* и *cvss_exploit*.

На рисунке 1 приведены гистограммы распределения оценок.

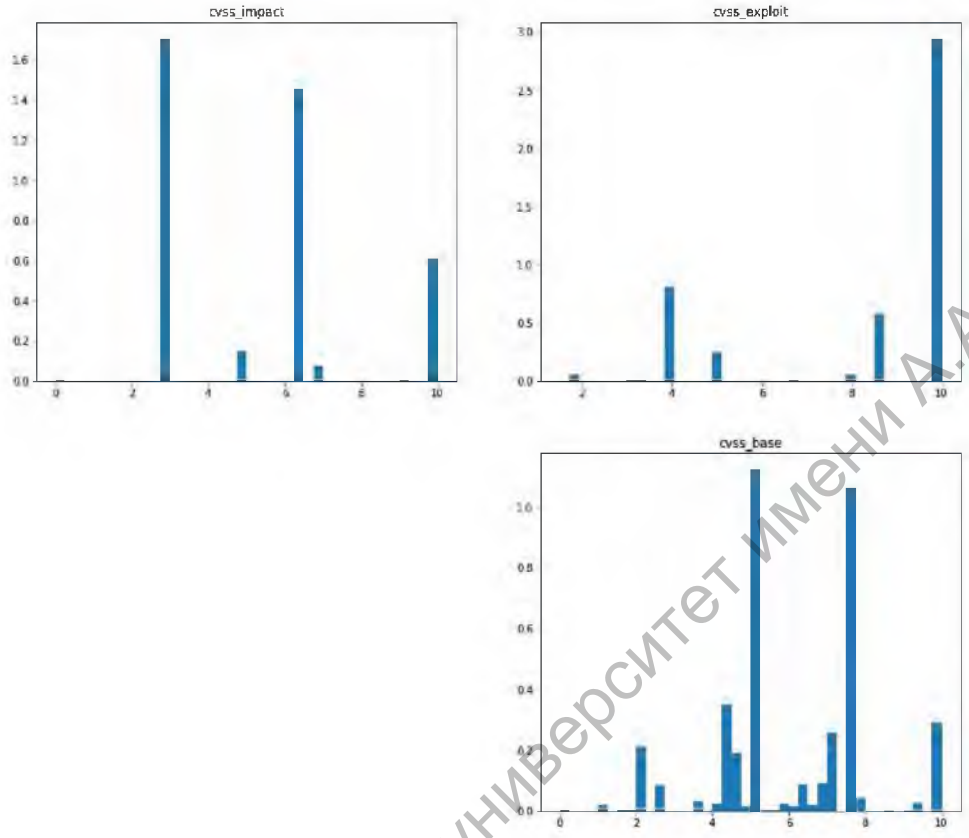


Рис. 1. Гистограммы распределения cvss_base, cvss_exploit, cvss_impact

Для проведения дальнейших исследований критерий оценок CVSS Base был выбран в качестве “главной оценки” степени критичности уязвимости как наиболее информативный. На рисунке 2 представлена гистограмма распределения оценок CVSS Base.

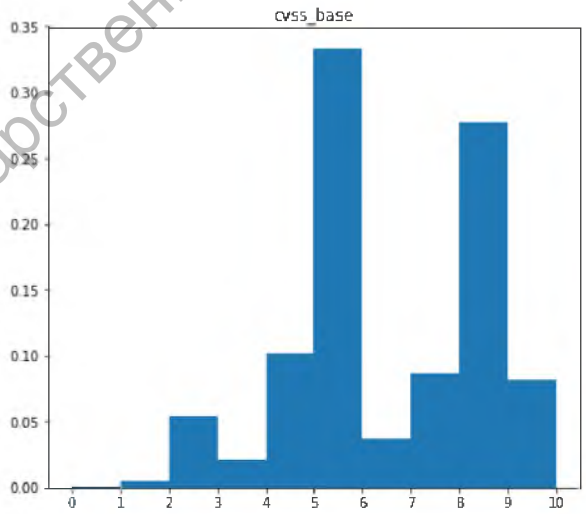


Рис. 2. Гистограмма распределения оценок CVSS Base

На гистограмме видно, что примерно половина всех оценок сгруппирована вокруг отметки в 4.5 балла, а остальная половина – вокруг 8. Таким образом составим два класса уязвимостей, описанных в таблице 2.

Таблца 2 – Классы уязвимости

Класс	Название	Правило соотнесения	Доля класса в датасете
0	Некритичные уязвимости	***CVSSBase<= 5***	51.635%
1	Критичные уязвимости	***CVSSBase> 5***	48.365%

Как видно из столбца 4 таблицы 2, доля классов практически одинаковая, поэтому набор данных не нуждается в предварительной нормализации.

Постановка задачи

Сформулируем задачу следующим образом.

Необходимо смоделировать систему, которая смогла бы автоматически бинарно классифицировать входящее текстовое сообщение по следующим классам:

- ***класс 0*** – описание уязвимости с низкой оценкой (от 0 до 5 баллов включительно);
- ***класс 1*** – описание уязвимости с высокой оценкой (от 5 до 10 баллов).

В качестве входных данных для обучения модели – текстовое описание уязвимостей из набора данных.

Таким образом, имеем задачу классификации с двумя классами: класс 0 – уязвимости с низкими оценками, класс 1 – с высокими оценками.

Выбор модели векторных представлений слов

Очевидно, что для обучения модели на текстовом описании необходимо некоторым образом получить возможность представить отдельные слова в виде некотором n-мерном векторном пространстве. Эту задачу решают так называемые модели векторных представлений слов.

Векторное представление слов (англ. *word embedding*) – общее название для различных подходов к моделированию естественного языка, направленных на сопоставление словам из некоторого словаря векторов небольшой размерности [5].

Существует несколько подходов для построения такого сопоставления.

Матрицы со-упоминаний слов в контексте других слов

Первые подходы основывались на построении матрицы совместных упоминаний слов в тексте и затем понижения ее размерности каким-либо из методов. Данный подход основан на предположении о том, что слова, встречающиеся в подобных контекстах, как правило, имеют сходные значения. Таким образом, использование статистики совстречаемости слов является естественным выбором для встраивания подобных слов в общее векторное пространство [6].

Суть данного подхода состоит в следующем. По корпусу текстов D и словарю T строится матрица со-упоминаний слов $X_{|T|x|T|}$.

Фиксируется размер контекстного окна вокруг каждого слова.

Контекстное окно – это число, отражающее количество контекстных слов слева и справа от текущего слова в тексте. Все слова из данного окна будут считаться упоминаниями по отношению к текущему слову. Порядок слов при этом не играет роли.

Матрица $X_{|T|x|T|}$ заполняется поэлементно. По всему корпусу текстов подсчитывается сумма числа упоминаний слова T_i в контексте слова T_j. Данная сумма затем нормализуется по числу корпусов текстов и становится элементом матрицы x_{ij}. В зависимости от

задачи применялись и иные варианты формирования матрицы (например, по количеству документов, хоть раз содержащих пару слов (T_i, T_j) или по количеству документов, хоть раз содержащих (T_i, T_j) в окне [7]).

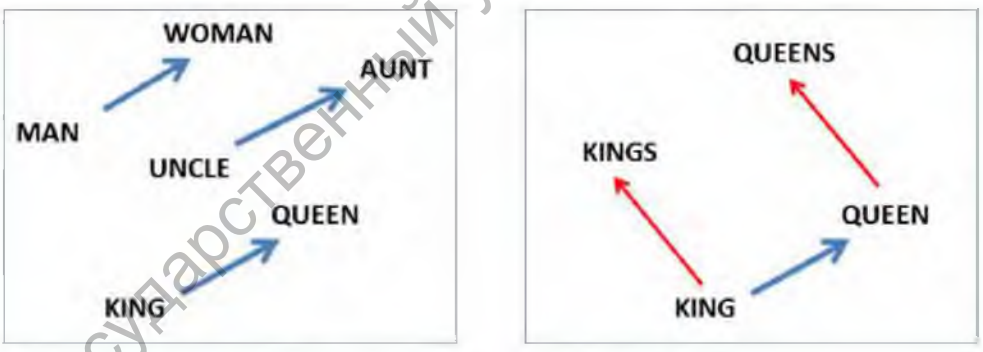
На выходе получалась матрица размером |T|x|T|. Она сильно зависела от размера словаря и являлась практически неприменимой на практике. Для устранения данного недостатка – понижения размеров матрицы – применялись методы линейной алгебры, например, сингулярное разложение:

$$X = USV^T.$$

Методы понижения размерности матрицы давали возможность строить статистику на основе словарей и корпусов текстов огромных размерностей. Однако на выходе получалось относительно низкое качество получаемых – представлений [6].

Word2vec – алгоритм, использующий машинное обучение для предсказания контекста слова

Следующим развитием подходов к векторизации слов является результат работы Томаша Миколова – алгоритм *Word2vec* [7]. Алгоритм строит векторные представления, применяя нейронную сеть в своей основе. Основная идея алгоритма заключается в том, что модель (нейронная сеть) обучается, учитывая контекстное окно вокруг каждого слова. Словам в модели сопоставлен уникальный вектор. Данный вектор изначально является вырожденным – все его элементы, кроме одного, равны 0, а оставшийся равен 1. Он изменяется в процессе обучения и будет являться выходным вектором модели после всех эпох ее обучения. Изменения вектора происходят в конце каждой эпохи обучения, минимизируя методом обратного распространения вероятность ошибки встречи текущего слова с учетом окружающих его слов из контекстного окна. Поэтому похожие по смыслу слова будут иметь похожие векторные представления, чего не удавалось достичь при помощи матрицы со-упоминаний слов и прочих статистических методов. Пример из работы автора алгоритма представлен на рисунке 3.



(Mikolov et al., NAACL HLT, 2013)

Рис. 3. Пример соотнесения векторов некоторым словам

На рисунке 3 видно, что векторы, относящиеся к словам “женщина” и “мужчина” соотносятся между собой идентично векторам, обозначающим слова “тетя” и “дядя”. Иными словами, вектора в данной модели несут семантический смысл контекста того слова, к которому относится вектор. Еще один пример автора над векторами, соответствующий словам “king”, “man”, “woman”:

[king] + [woman] – [man] = [queen]

Векторные преобразования над векторами word2vec частично сохраняют смысл исходного контекста, позволяя использовать векторы более естественным образом.

Алгоритм GloVe-развитие Word2vec

Алгоритм GloVe (GlobalVectors) был разработан исследовательской группой Стэнфордского университета в 2015 г. [8]. Он является сочетанием идеи Word2vec и подхода, основанного на формировании матрицы со-вхождений. Алгоритм работает по тому же принципу, что иWord2vec. Однако отличие состоит в том, что вместо функции минимизации ошибки предсказания контекста в GloVeдля обучения модели используется матрица со-вхождений слова, аналогичная той, что использовалась в ранних подходах векторизации слов.

Word2vec пытается предсказать контекст каждого слова, в то же время GloVe учится на основе предварительно построенной матрицы со-вхождений слова в контекст каждого другого слова и предсказывает частотность встречаемости слова в данном контексте. Выходные векторы, полученные при помощи GloVe, сохраняют информацию о глобальной статистике каждого слова, а не только (как в Word2vec) информацию из контекста локального окна фиксированного размера.

GloVeявляется развитием Word2vec, используя тот же подход и нивелируя его недостатки. Поэтому для решения поставленной задачи нами далее будут использоваться векторные представления слов, сформированные по алгоритму GloVe.

Выбор набора подготовленных векторных представлений GloVe

На странице проекта GloVe присутствует несколько предварительно подготовленных авторами наборов векторных представлений слов, словарей по типу “слово-вектор”, обученных на четырех корпусах текстов [9]. Они представлены в таблице 3.

Таблица 3 – Предварительно подготовленные векторные представления слов

Источник	Общее количество слов	Количество слов в словаре	Размерность векторов
Wikipedia 2014 + Gigaword 5	6.000.000.000	400.000	50, 100, 200, 300
Common Crawl	42.000.000.000	1.900.000	300
Common Crawl	840.000.000.000	2.200.000	300
Twitter	27.000.000.000	1.200.000	25, 50, 100, 200

В исходном наборе данных всего около 100.000 записей об уязвимостях. Также известно, что описания уязвимостей составлены с использованием только литературных слов и специфичных терминов. Большинство терминов имеется в энциклопедии Википедия вместе с их описанием. Поэтому для проведения исследования мы остановились на использовании словаря векторных представлений слов, построенных на основе всех англоязычных статей Википедии.

Мы остановились на использовании 50-размерных векторов с целью минимизации времени обучения. В разделе “Использование словарей больших размерностей” приводится сравнение качества модели при использовании 50, 100, 200 и 300-размерных векторов.

Исходный код построенной модели на Python

В качестве предиктивного алгоритма было принято решение использовать конволюционную нейронную сеть, так как данный тип нейросетей обычно показывает хорошие результаты в задачах обработки естественного текста. В результате была построена

многослойная нейронная сеть с двумя выходами. На вход нейросети подается текстовое описание уязвимости, которое преобразуется в вектор в соответствии с векторным представлением GloVe. Далее преобразованные данные проходят три конволюционных слоя (Conv1d), после каждого из них – следует обобщающий (MaxPooling) слой. С целью уменьшения переобучения затем идут три полносвязных (Dense) слоя в сочетании с исключающими (Dropout) слоями. Сеть обучается, используя оптимизатор Adam и кросс-энтропийную функцию в качестве функции потерь. Весь исходный код доступен по ссылке: https://github.com/teacherlex/cve_vulns_classifier

Для построения модели нейронной сети использовались библиотеки Tensorflow [10] и Keras [11]. В Листинге 1 представлен исходный код программы обучения нейронной сети.

```
dropout = 0.4
x = Conv1D(128, 5, activation='relu')(embedded_sequences)
x = SpatialDropout1D(dropout) (x)
x = MaxPooling1D(5)(x)
x = Conv1D(128, 5, activation='relu')(x)
x = SpatialDropout1D(dropout) (x)
x = MaxPooling1D(5)(x)
x = Conv1D(128, 5, activation='relu')(x)
x = SpatialDropout1D(dropout) (x)
x = MaxPooling1D(35)(x)
x = Flatten()(x)
x = Dense(128, activation='relu')(x)
x = Dropout(dropout) (x)
x = BatchNormalization() (x)
x = Dense(128, activation='relu')(x)
x = Dropout(dropout) (x)
x = BatchNormalization() (x)
x = Dense(128, activation='relu')(x)
x = Dropout(dropout) (x)
x = BatchNormalization() (x)
preds = Dense(2, activation='softmax')(x)

model = Model(sequence_input, preds)
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['acc',auc])

model.fit(x_train, y_train,
batch_size=128,
          epochs=20
,validation_data=(x_val, y_val))
```

Листинг 1. Исходный код программы

Оценка качества модели

Всего в базе данных уязвимостей NVD имеется 94.949 записей. Обучение происходит на 75.960 из них (эту выборку далее будем называть *тренировочной* или *обучающей*).

Для проверки качества построенной модели используется *валидирующая выборка* – случайным образом отобранные записи из исходного набора данных. Нами было отобрано 20% записей исходной базы данных, что составляет 18.989.

Валидирующая выборка предназначена для оценки построенной модели. Оценка производится по различным критериям, так называемым *метрикам*. Мы использовали следующие метрики оценки качества модели:

• ***Точность предсказания*** (*англ. "accuracy"*) – соотношение количества верных предсказаний классификатора к размеру обучающей выборки [12]. Точность является наиболее простой метрикой в области анализа данных и вычисляется по формуле

$$Accuracy = \frac{T}{N},$$

где T – количество верных предсказаний, а N – общее количество примеров.

• ***Площадь под ROC-кривой*** (*англ. "ROCcurve", receiver operating characteristic curve*) – аггрегированная характеристика качества классификации, не зависящая от соотношения цен ошибок. Чем больше значение площади по ROC-кривой, тем "лучше" модель классификации [13]. Данную метрику также сокращенно называют AUC-ROC (AUC – AreaUnderCurve, площадь под кривой).

Мы провели 20 эпох обучения нейронной сети. На рисунке 4 ниже представлены графики изменения метрик для тренировочной и валидирующей выборок.

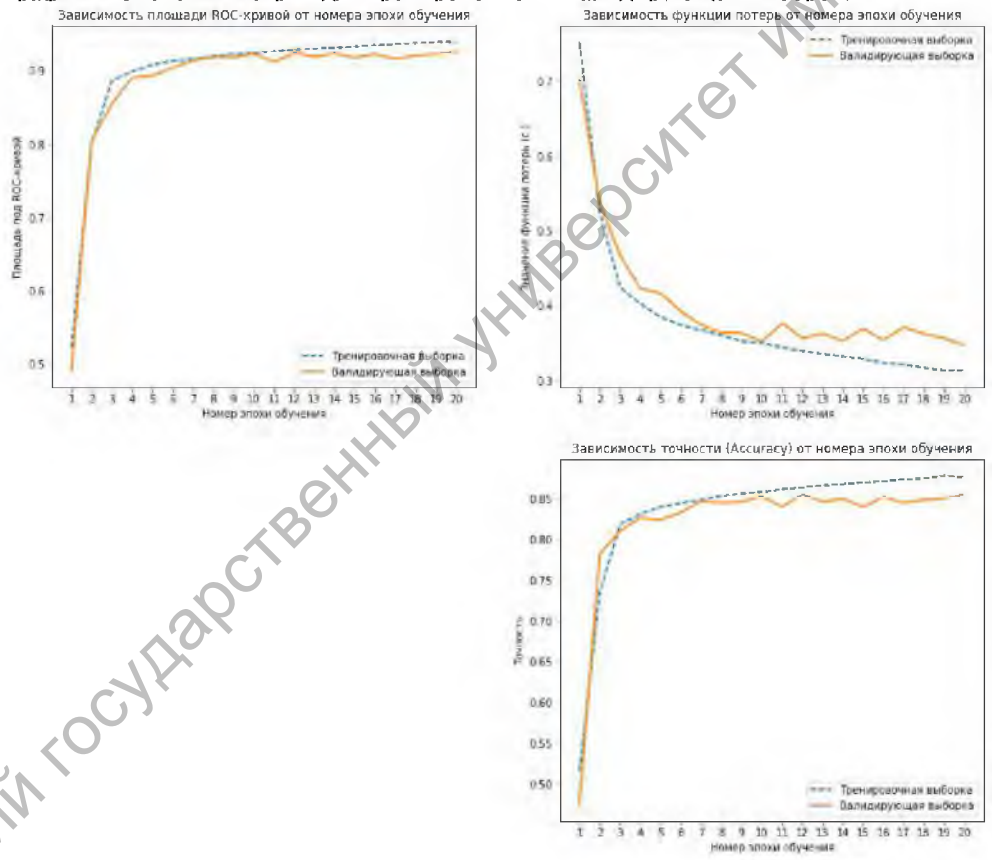


Рис 4. Графики изменения метрик в зависимости от эпохи обучения

Все метрики стабилизируются на валидирующей выборке на 5-6 эпохах. Максимальный показатель точности достигает значения 85.52%, а площади под ROC-кривой – 92.74%.

Начиная с 6 эпохи обучения, происходит небольшое устойчивое увеличение разницы в показателях между валидирующей и тренировочной выборкой. Это означает,

что нейросеть медленно начинает подстраиваться под тренировочную выборку, что приводит к переобучению при росте номера эпохи. Поэтому в практических задачах имеет смысл остановить обучение на 6 эпохе.

Использование классических методов машинного обучения: случайный лес, наивный байесовский классификатор

Для сравнительного анализа качества модели полезно попытаться применить и другие классификаторы (методы машинного обучения для задачи классификации). Были применены следующие алгоритмы: случайный лес (RandomForest [14]) для 1500 деревьев в ансамбле и наивный байесовский классификатор (naiveBayesclassifier [15]). Сравнительные результаты показателей качества представлены в таблице 4.

Таблица 4 – Сравнение результатов работы различных методов машинного обучения

Метод	Точность предсказания	Площадь по ROC-кривой
Наивный байесовский классификатор	53.66%	66.9%
Случайный лес	79.26%	87.2%
Нейронная сеть	85.52%	92.74%

Исходя из представленных данных, можно сделать вывод о том, что наивный байесовский классификатор справился с задачей значительно хуже оставшихся алгоритмов. Случайный лес отстает по показателям на 5–6%, причем рост количества деревьев не дает заметного роста качества модели. Предположительно, построенная модель нейронной сети из-за своей природы сумела найти скрытые взаимосвязи между словами в предложении и степенью критичности уязвимости и поэтому показывает лучшие показатели. Поэтому далее будем рассматривать только модель, построенную на основе нейронной сети.

Сравнение размерностей векторов GloVe для последующего практического применения

На реальную производительность системы будет также играть размерность векторов GloVe (50, 100, 200 или 300), т. к. вместе с размерностью будет соответствующе расти объем выборки и, следовательно, время обучения каждой эпохи. Имеет смысл сравнить различные размерности векторных представлений на практике, чтобы выбрать наиболее подходящее из них для использования в практических задачах.

Мы протестировали модель, используя при обучении различные векторные представления GloVe (50-, 100-, 200- и 300-размерные). Результаты приведены на рисунке 5.

Из графиков видно, что на начальных эпохах обучения метрики качества (точность, AUC-ROC, значение функции потерь) значительно выше у моделей, использующих более высокую размерность векторов. Однако с ростом эпох результаты выравниваются, и далее каждая модель показывает сравнительно одинаковые результаты.

При этом время обучения каждой эпохи в зависимости от размерности отличается значительно. Исходя из этого, можно сделать следующий вывод: при практическом применении имеет смысл использовать вектора минимальной размерности (50), чтобы уменьшить время обучения следующей эпохи на обновленных данных.

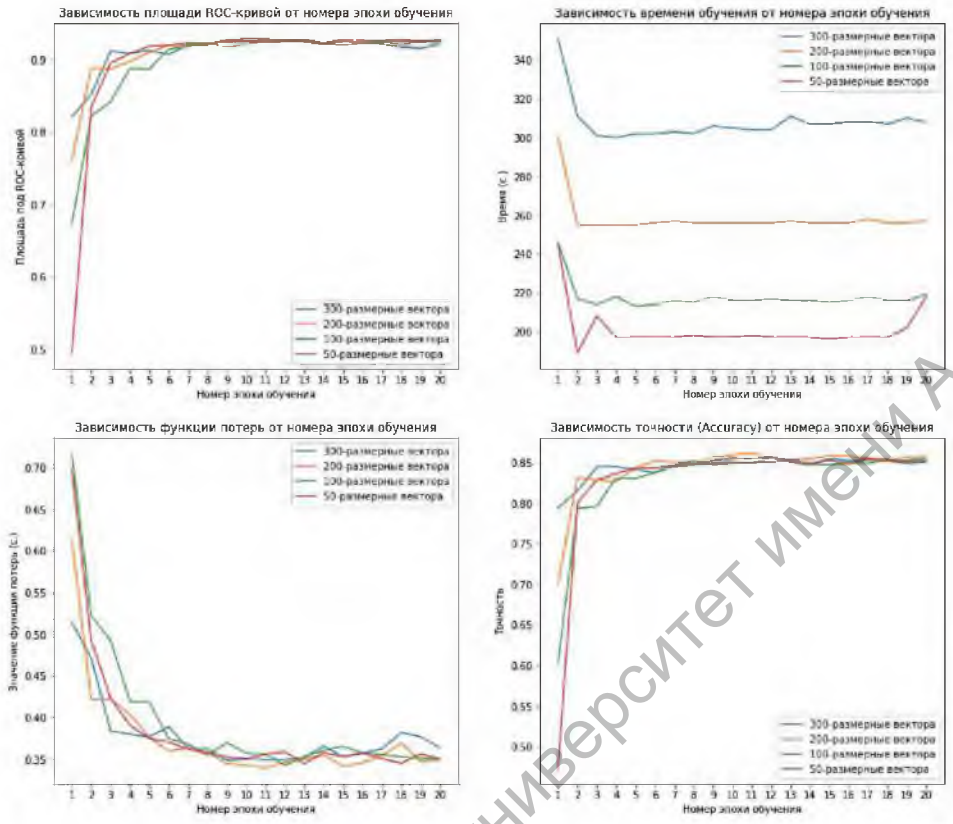


Рис. 5. Графики зависимостей показателей обучения от номеров эпохи обучения для различных размерностей векторов GloVe (50, 100, 200, 300)

Варианты практического применения модели. Оценка новых уязвимостей и новостных записей

Модель можно применять к только появившимся уязвимостям, которым еще не выставлена оценка NVD, чтобы предварительно выяснить их степень критичности до выставления оценки экспертами.

Также можно применить модель к новостным источникам, связанным с информационной безопасностью. К примеру, можно в первую очередь показывать офицеру безопасности "критичные" новостные записи, а только лишь потом – некритичные. Таким образом, в случае появления ранее неизвестной критичной уязвимости у офицера безопасности будет больше времени предпринять возможные меры по предотвращению ее эксплуатации в организации.

Оценка критичности баг-репортов и коммитов на www.GitHub.com

Также существует актуальная задача предварительной автоматической оценки нового кода в OpenSource-проектах на www.GitHub.com. В этом случае данными на вход модели могут служить текстовое описание из следующих GitHub-источников, связанных с уязвимостями [16] (должны быть помечены тегом #vulnerability_relatives):

- Отдельные коммиты
- Баг-репорты
- Пулл-реквесты

Модель может выполнять функцию выставления предварительной оценки (например, критичен ли баг-репорт с точки зрения безопасности или же нет), и далее ревьювер кода может либо согласиться с данной оценкой, либо нет, таким образом выполняя этап дообучения модели на новой порции данных. Верная предварительная оценка позволит ревьюверу концентрироваться в первую очередь на проверке кода, связанного с более критичными уязвимостями, и лишь во вторую очередь – с менее опасными.

Заключение

В статье описана модель машинного обучения, предсказывающая критичность той или иной уязвимости, основываясь лишь на ее текстовом описании. В качестве звена, преобразующего текст в векторное пространство, используется 50-мерный словарь проекта GloVe, а в качестве основного обучающегося алгоритма – многослойная конволюционная нейронная сеть. Наилучшая из построенных нами моделей достигла показателя 92.74% по метрике площади под ROC-кривой. Также выполнено сравнение словарей GloVeразличных размерностей (50, 100, 200, 300), обоснован выбор векторов размерности 50. Построенную модель можно использовать в дальнейшем для оценки любого текста, а не только являющегося непосредственно описанием уязвимости. К примеру, можно предварительно классифицировать степень критичности баг-репортов по их описанию на ресурсах для управления контроля версиями, что позволит автоматизировать часть работы с баг-репортом.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. National Vulnerability Database [Электронный ресурс] / National Vulnerability Database. – Режим доступа: https://cve.mitre.org
2. Общая система оценки уязвимостей CVSS [Электронный ресурс] / BISExpert. – Режим доступа: http://bis-expert.ru/blog/5345/43124
3. Vulnerability Details: CVE-2008-4250 [Электронный ресурс] / CVE Details. – Режим доступа: https://www.cvedetails.com/cve/CVE-2008-4250/
4. Anticipating Cyber Vulnerability Exploits Using Machine Learning [Электронный ресурс] / Semantic Scholar. – Режим доступа:
5. https://pdfs.semanticscholar.org/8394/ec608f6167d6c79ec2c495595249b439d959.pdf
6. ***Levy, Omer; Goldberg, Yoav.*** Linguistic Regularities in Sparse and Explicit Word Representations // Proceedings of the Eighteenth Conference on Computational Natural Language Learning, Baltimore, Maryland, USA, June. Association for Computational Linguistics. – 2014.
7. Векторные представления слов [Электронный ресурс] / Профессиональный информационно-аналитический ресурс, посвященный машинному обучению, распознаванию образов и интеллектуальному анализу данных. – Режим доступа:
8. http://www.machinelearning.ru/wiki/images/b/b3/Word2Vec.pdf
9. ***Mikolov T., Chen K., Corrado G, Jeffrey D.*** Efficient estimation of word representations in vector space. InICLR Workshop Papers, 2013a.
10. ***PenningtonJ., Socher R., Manning C.*** GloVe: Global vectors for word representation. InProceedings of EMNLP-2014. Doha, Qatar, October2014, p. 1532–1543.
11. Global Vectors for Word Representation [Электронный ресурс] / The Natural Language Processing Group at Stanford University. – Режим доступа: https://nlp.stanford.edu/projects/glove/
12. ***Abadi M. et al.*** Tensorflow: A system for large-scale machine learning //12th Symposium on Operating Systems Design and Implementation. – 2016. – P. 265–283.
13. ***Chollet F. et al.*** Keras. – 2015.
14. Scikit-learn: Machine learning in Python / F. Pedregosa [et al.] // J. of Machine Learning Research. – 2011. – No. 12. – P. 2825–2830.
15. ROC Graphs:Notes and Practical Considerations for Researchers [Электронный ресурс] / George Mason University. – Режим доступа: http://binf.gmu.edu/mmasso/ROC101.pdf

16. ***Breiman L.*** / Random Forests // Machine Learning journal. – 2001. – Vol. 45, no. 1. – P. 5–32.
17. ***Pedro D., Pazzani M.*** / On the optimality of the simple Bayesian classifier under zero-one loss// Machine Learning. – 1997. – Vol. 29. – P. 103–137.
18. Using Machine Learning to Identify Security Issues in Open-Source Libraries [Электронный ресурс] / GitHub.com. – Режим доступа: https://asankhaya.github.io/pdf/Using-machine-learning-to-identify-security-issues-in-open-source-libraries.pdf

Поступила в редакцию16.09.2019 г.
Контакты: e-mail: aliexei.doronin@gmail.com (Доронин Алексей Константинович)

Doronin A., Lipnitsky V. BUILDING MACHINE LEARNING MODEL FOR THE CVE VULNERABILITY SEVERITY CLASSIFICATION PROBLEM.

The article discusses the use of machine learning methods in combination with the algorithm of word representation in the multidimensional vector space GloVe for the problem of predicting the criticality of vulnerability based only on its textual description. The NVD vulnerability database is used as a dataset for analysis and training. The article presents the analysis of vulnerability records, vulnerability assessment mechanism, on the basis of which the choice of features for model training is justified. Various approaches and methods for constructing vector representations of words are described, the choice of vectors constructed by the GloVe algorithm is justified. A comparative analysis of the model on sets of GloVe vectors of different dimensions is also given, conclusions about the use of 50-dimensional GloVe vectors are drawn. Several possible further practical applications of the trained model are given.

Keywords: machine learning, convolution neural networks, data analysis, vulnerability of computer systems, assessment of vulnerability severity, vector representation of words, text processing.