Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

НА ТЕМУ:

«Метод фильтрации спам-сообщений электронной почты на основе нейронной сети и метода опорных векторов»

Студент ИУ7-81Б (Группа) _____ (Подпись, дата) Д. В. Яковлев (И. О. Фамилия)

Руководитель ВКР _____ (Подпись, дата) З. Н. Русакова (И. О. Фамилия)

Нормоконтролер _____ (Подпись, дата) _____ (И. О. Фамилия)

2024 г.

РЕФЕРАТ

Расчетно-пояснительная записка 75 с., 21 рис., 4 табл., 15 источн., 1 прил.

КЛЮЧЕВЫЕ СЛОВА: СПАМ, ЭЛЕКТРОННАЯ ПОЧТА, МАШИНА ОПОРНЫХ ВЕКТОРОВ, НЕЙРОННАЯ СЕТЬ, ПЕРЦЕПТРОН, БЕЗОПАСНОСТЬ

Объектом исследования является метод фильтрации спам-сообщений электронной почты на основе нейронной сети и метода опорных векторов.

Целью работы являлась разработка метода фильтрации спама на основе нейронной сети и метода опорных векторов.

Для достижения поставленной цели были решены следующие задачи:

— проведен анализ предметной области.

— разобраны основные методы классификации.

— спроектировано программное обеспечение.

— реализовано программное обеспечение.

— проведено исследование на тему конкурентоспособности реализованного метода и известных аналогов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Одним из направлений исследований в области защиты информации является разработка методов и алгоритмов фильтрации потока электронной почты. В последнее время электронная почта стала одним из наиболее распространенных средств связи, управления и бизнеса. Она является достаточно совершенной в техническом отношении и недорогой альтернативой привычным средствам связи.

Вместе с развитием электронной почты увеличивается и количество угроз ее нормальному функционированию. Наиболее серьезной и важной проблемой стал так называемый снам, то есть нежелательные массовые рассылки сообщений, в основном рекламного характера. В отчете «Лаборатории Касперского», в 2023 году доля спама составляет 46% писем по всему миру и 47% писем в России [1].

На сегодняшний день разработан ряд технологий построения сервисов, фильтрующих спам, можно разделить на настраиваемые вручную и интеллектуальные. Настраиваемые вручную фильтры работают напрямую от заданных пользователем настроек, где пользователь указывает либо нежелательные адреса, то есть по так называемому «черному списку», либо разрешенные адреса, то есть по «белому списку». Однако данные фильтры не только малоэффективны, но еще и требуют постоянной настройки от пользователя.

В свою очередь фильтры, построенные с использованием технологий искусственного интеллекта, обучаются только на начальном этапе, и в дальнейшем обучаются самостоятельно, существенно снижая нагрузку на пользователя.

Самым распространенным на сегодняшний день является фильтр, основанный на наивном байесовском подходе, в котором предполагается, что различные термы сообщения независимы друг от друга. Для повышения эффективности данного фильтра необходимо учитывать семантические связи между термами, что требует привлечения методов семантического анализа и существенно повышает нагрузку на систему и увеличивает время работы самого фильтра, при незначительном повышении эффективности фильтрации.

Другим подходом, получающим в последнее время все большее распространение, является использование нейросетей. Преимущество нейросетевого

подхода перед наивным байесовским состоит в том, что не делается никаких предварительных предположений о характере нежелательных сообщений, а семантические связи учитываются автоматически. Наибольшее количество разработок связано с построением фильтра на основе многослойного перцептрона. Однако такой подход встречается с рядом трудностей, связанных с выбором пороговых значений, которые задаются произвольно в некотором интервале. Эффективность фильтра существенно зависит от выбора порогового значения. При этом пороговое значение требует постоянной подстройки под изменяющийся характер нежелательных сообщений. Также малоисследованным остается вопрос использования других нейросетей, хорошо зарекомендовавших себя в задачах распознавания образов, частным случаем которых является фильтрация спама. Таким образом, развитие нейросетевого подхода применительно к фильтрации нежелательных сообщений является актуальной задачей.

1 Аналитический раздел

В данном разделе проведен анализ предметной области фильтрации спам-сообщений электронной почты. Приведен разбор методов классификации, а также рассмотрен анализ при помощи нейронной сети и метода опорных векторов.

1.1 Анализ предметной области

Спам – сообщения (в данном случае, электронные письма), массово рассылаемые людям, не дававшим согласие на их получение, а также любые многократно повторяющиеся сообщения не несущие смысловой нагрузки, либо теряющие её в результате многократного повторения (сюда входят как сообщения рекламного, так и не рекламного характера) [2].

Также, значительное количество спама содержит вирусы во вложении. Основываясь на этом, спам можно разделить на две категории: спам без вложения, и спам с вложением, то есть сообщение с прикрепленным к нему файлом.

За основу возьмем статистику полученных спам-сообщений на корпоративный почтовый сервер, полученных за две недели. На рисунке 1.1 изображено количество спама обеих категорий, ось X – соответствующий день, ось Y – количество спам-сообщений, полученных пользователями.

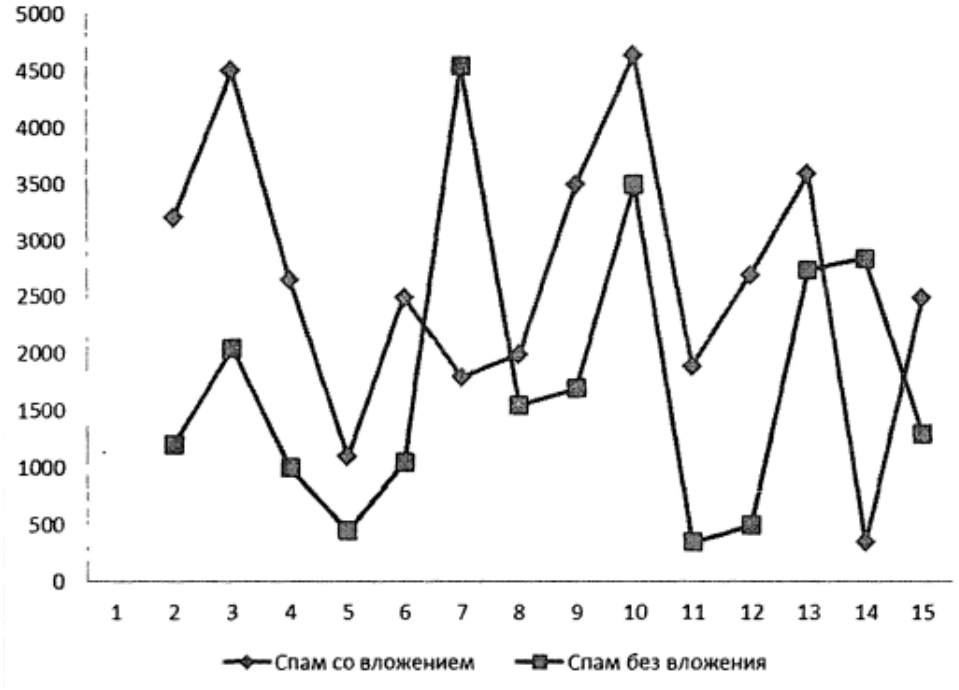

Рисунок 1.1 – Спам во входящем трафике [1]

По результатам сбора данных, в среднем в день пользователи получали 1872 спам-сообщения с вложениями в день, и 2722 спам-сообщений без вложенных файлов.

Спам без вложений это обыкновенные текстовые сообщения. Их можно классифицировать как сообщения, которые содержат только текст, где может также может быть включен URL или гиперссылка на сайт. Спам, содержащий только текст в основном связан с мошенничеством и не имеет массового характера, в свою очередь, спам с URL или гиперссылками в основном связан с рекламой и гораздо больше распространен.

Из всего полученного спама более 50% является спам с вложениями. Подобные сообщения содержат комбинацию изображений, текста, URL, гиперссылок, и исполняемых файлов. Размер данных сообщений существенно больше спам-сообщений без вложений. Данные спам-сообщения, в свою очередь, можно разбить на четыре основные категории:

1. спам, содержащий файлы с изображениями (GIF) и текст;
2. спам, содержащий изображение, текст и URL;
3. спам, содержащий только изображения с гиперссылкой;
4. спам, содержащий вирусы или троянские программы.

Наиболее опасной является категория номер 4. Цель подобных сообщений – распространение вирусов, а также попытка установки программ для DDoS атак на сервер и сеть. Подобные атаки используют всю пропускную способность сети и её ресурсов, в результате чего медленно доставляется почта и возникает вероятность отказа в обслуживании.

Ущерб, наносимый спамом, состоит из трех компонентов:

1. трафик. Входящий почтовый трафик, как правило, оплачивается получателем сообщения. Это особенно актуально, когда подключение к Интернету производится по телефонным линиям. Для крупных компаний, подключенных по выделенной линии, финансовый ущерб из-за большого объема пересылаемой почты и, вместе с тем, большого объема нежелательных сообщений, является существенным. Проведем анализ влияния спама S на доступность информации 1.2. Видно, что если в определенный момент времени t_0 объём поступающей полезной электронной корреспонденции на почтовые ящики компании и спам-сообщений S превышает пропускную способность канала связи , соединяющего организацию с Интернет-провайдером, следовательно, некоторое количество электронных сообщений L, в том числе и полезных, будет получено с опозданием или потеряно. Тем самым, возникает очередь из запоздавших сообщений, что приводит к еще большей временной задержке при доступе к требуемой информации. Если в некоторый момент времени t_i общий объём и S станет меньше связи компании с провайдером, то пользователи смогут своевременно получать доступ к сообщениям.

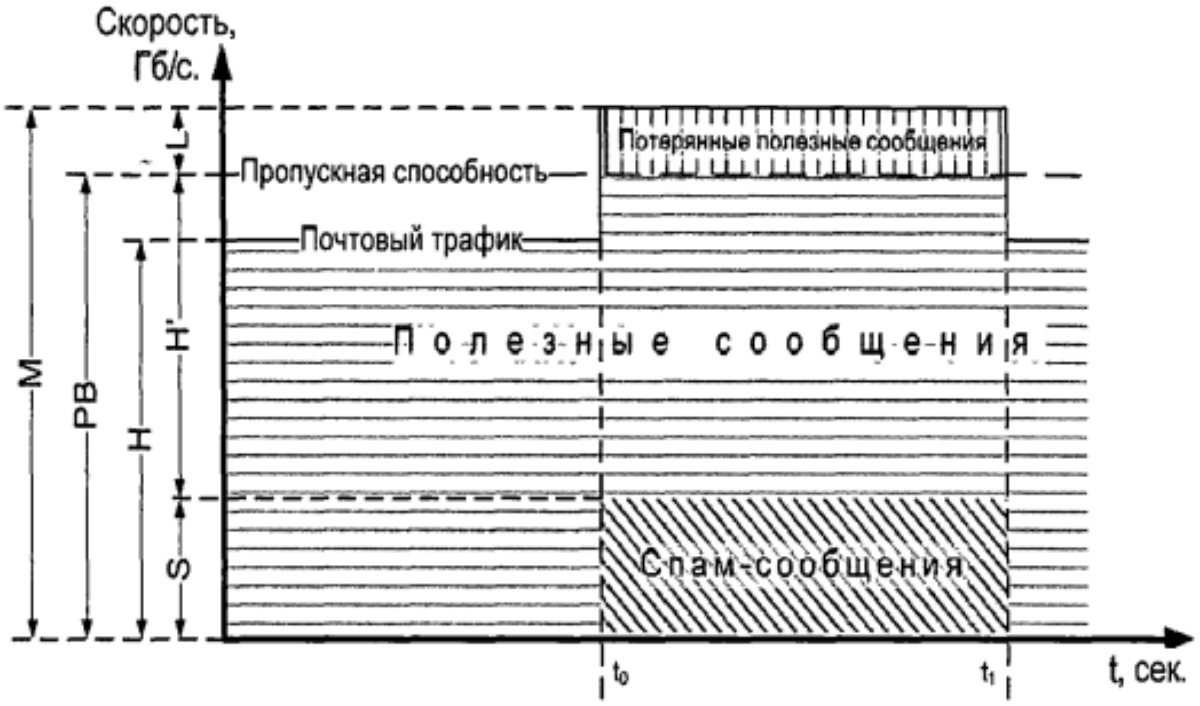

Рисунок 1.2 – Нарушение доступности [1]

Также, спам-сообщения нарушают целостность информации, что изображено на 1.3. Допустим, что в некоторый момент времени t_0 пользователь, выполняющий ручную классификацию электронной почты, ошибочно отфильтрует часть полезных электронных писем LH, а также пропустит часть спама RS. К подобным последствиям также может привести применение малоэффективных средств автоматизированной фильтрации входящих сообщений.

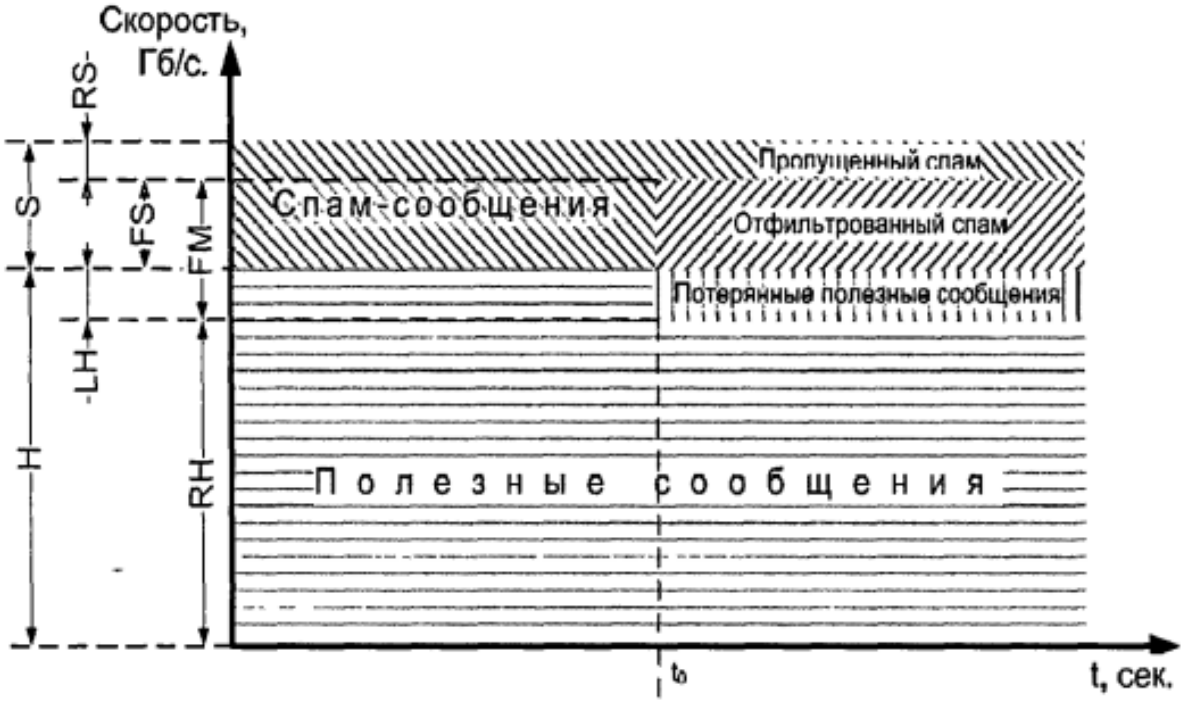

Рисунок 1.3 – Нарушение целостности [1]

2. Потери рабочего времени. По результатам исследования «Лаборатории Касперского» сотрудники, которые получают до 30 внешних писем в день, тратят на разбор спама около 5 часов в год. Те, кому приходит от 30 до 60 писем, теряют на этом до 11 часов в год, а те, кому от 60 до 100 писем, – 18 часов [3]. Умножая это время на общее количество персонала, выходит огромное количество потерянных часов.

3. Ущерб безопасности системы. Спам наносит ущерб не только в виде затрат рабочего времени и оплаты лишнего трафика. Спам-сообщения довольно часто являются переносчиком вредоносных программ, поскольку рассылаются с вложениями в виде исполняемых файлов которые могут быть вирусами. Уже предпринимались попытки организации спам рассылок, используя уязвимости в безопасности почтовых серверов, то есть спамерские атаки проводиться совместно с атаками хакеров на узлы сети Интернет. В случае если подобная атака проходит успешно, во-первых, возникнет значительная нагрузка на атакуемые узлы сети, которая как минимум замедляет, а в самом неудачном варианте - блокирует их работу, а во-вторых, значительный ущерб наносится репутации компании, которая является собственником сервера, использованного для рассылки спама.

По борьбе со спамом есть два принципиально разнмых метода. Первый метод направлен на предотвращение рассылки спама с помощью, например, установления цены отправки сообщения или путем блокирования или ограничения доступа к почтовым серверам. Второй метод направлен на обнаружение и удаление отправленного спама с помощью различных методов фильтрации.

Самым очевидным способом предотвращения рассылок спама является закрыть все открытые реле в Интернет, а также укрепить протокол SMTP. Это заставит почтовые программы для рассылки спама использовать своих собственных Интернет-провайдеров, но в этом случае провайдеры будут блокировать их. Однако, этот подход идет в разрез с открытой философией Интернета и создает другие угрозы для неприкосновенности частной жизни в Интернете. Более того, закрыв все открытые реле будет недостаточно, так как спамеры теперь переходят на использование открытых прокси-серверов или даже использование взломанных компьютеров.

Одним из интересных и, возможно, наиболее целесообразных подходов, является предложение остановить нежелательные массовые рассылки с помощью экономических решений. Существует две основные категории экономических решений.

Первая, это сделать невыгодной рассылку спама за счет увеличения затрат вычислительных ресурсов спамера, при отправке каждого сообщения. При данном методе борьбы с массовыми рассылками, отправителю предварительно необходимо вычислить умеренной сложности функцию. Эта функция вызывается функция цены.

Вторая категория экономических решений в борьбе со спамом, это увеличение финансовых затрат на отправку сообщений, т.е. требовать небольшую плату за отправку каждого электронного сообщения. Система основана на том, что пользователю необходимо заплатить небольшую сумму, прежде чем он сможет читать сообщения приходящие на некоторых каналах.

К методам, направленным на обнаружение и удаление спама, относятся спам-фильтры. В общем, спам-фильтр можно определить как приложение, которое классифицирует входящие электронные письма как спам или легитимные сообщения, используя определенный метод классификации.

1.2 Разбор методов классификации

В данном разделе будут рассмотрены основные методы классификации для решения задачи фильтрации спам-сообщений, такие как логистическая регрессия, случайный лес и смешанный алгоритм фильтрации, основанный на методе опорных векторов (SVM) и нейронной сети.

1.2.1 Логистическая регрессия

Логистическая регрессия — это один из наиболее популярных алгоритмов классификации. Он основан на логистической функции, которая преобразует входные данные в вероятности принадлежности к классам. Логистическая регрессия может быть применена как для бинарной классификации, так и для многоклассовой классификации.

Суть логистической регрессии заключается в ее математической основе, в первую очередь основанной на логистической функции, также известной как сигмовидная функция. Эта S-образная кривая отображает любое действительное число в значение от 0 до 1, что делает ее исключительно подходящей для моделирования вероятностных распределений двоичных результатов. Таким образом, логистическая модель оценивает вероятность того, что данные входные данные принадлежат к определенной категории, что имеет решающее значение для задач классификации [4].

Делается предположение о том, что вероятность наступления события $y = 1$ равна:

$$\mathbf{P}\{y = 1|x\} = f(z), \tag{1}$$

где

$$z = \theta^T x = \theta_0 + \theta_1 x_1 + ... + \theta_n x_n, \tag{2}$$

x и θ – векторы-столбцы значений независимых переменных $1, x_1, ..., x_n$ и параметров (вещественных чисел) $\theta_0, \theta_1, ..., \theta_n$ соответственно, а $f(z)$ – логистическая функция (также называемая сигмоидой):

$$f(z) = \frac{1}{1 + e^{-z}}. \tag{3}$$

.

Для подбора параметров $\theta_0, ..., \theta_n$ требуется составить обучающую выборку, состоящую из наборов значений независимых переменных и соответствующих им значений зависимой переменной y. Формально, это множество пар $(x^{(1)}, y^{(1)}), ..., x^{(n)}, y^{(n)})$, где $x^{(i)} \in \mathbf{R}^n$ – вектор значений независимых переменных, а $y^{(i)} \in \{0, 1\}$ – соответствующее им значение y. Каждая такая пара называется обучающим примером.

Обычно используется метод максимального правдоподобия, согласно которому выбираются параметры θ, максимизирующие значение функции правдоподобия на обучающей выборке:

$$\hat{\theta} = argmax_\theta Ł(\theta) = argmax_\sigma \prod_{i=1}^{m} \mathbf{P}\{y = y^{(i)} | x = x^{(i)}\}. \quad (4)$$

Преимуществами данного метода являются:

— он не делает никаких предположений о распределении классов в пространстве функций;

— можно легко распространить на несколько классов;

— очень быстро классифицирует неизвестные записи;

— хорошая точность для многих простых наборов данных;

— меньшая склонность к переобучению.

Недостатками же являются:

— если количество наблюдений меньше количества функций, то может возникнуть переобучение;

— данный метод строит линейные границы;

— основное ограничение состоит в предположении о линейной зависимости переменной от независимых;

— данный метод можно использовать только для прогнозирования дискретных функций;

— ввиду линейной поверхности решения, нелинейные проблемы не могут быть решены данным методом;

— данный метод неустойчив к выбросам;

— требуется предварительная обработка данных;

— для данного метода требуется усреднение или отсутствие корреляции независимых переменных [5].

1.2.2 Случайный лес

Случайный Лес — это алгоритм машинного обучения, основанный на концепции деревьев решений и обеспечивающий прогностическую модель. Во время обучения создается множество деревьев решений. Он выводит класс, то есть режим классов (при классификации) или среднее предсказание (при регрессии) отдельных деревьев [6].

Данный метод работает следующим образом.

Этот метод случайным образом выбирает подмножество набора данных для каждого дерева, обеспечивая разнообразие деревьев и повышая надежность модели.

Строится решающее дерево, классифицирующее образцы данной подвыборки, причём в ходе создания очередного узла дерева из случайно выбранных признаков выбирается набор, на основе которых производится разбиение. Выбор наилучшего из этих признаков может осуществляться различными способами. В оригинальном методе Бреймана используется критерий Джини, применяющийся также в алгоритме построения решающих деревьев CART. В некоторых реализациях алгоритма вместо него используется критерий прироста информации. Данное дерево строится до полного исчерпания подвыборки, причем отсечения ветвей не происходит.

После этого классификация объекта строится путем голосования: каждое дерево относит объект к одному из классов, и побеждает тот класс, за которое проголосовало наибольшее число деревьев.

Преимуществами данного метода являются:

— данный метод устойчив к выбросам;

— высокая точность предсказания;

— не требует тщательной настройки параметров;

— одинаково хорошо обрабатывает как непрерывные, так и дискретные признаки;

— редко переобучается;

— не чувствителен к масштабированию;

— высокая параллелизуемость.

Недостатками же являются:

— отсутствие формальных выводов;

— метод работает хуже многих линейных методов, когда в выборке очень много разреженных признаков;

— алгоритм склонен к переобучению на зашумленных данных;

— если данные содержат группы коррелированных признаков, имеющих схожую значимость для метод, то предпочтение отдается небольшим группам перед большими;

— большой размер получающихся моделей, требуется $O(NK)$ памяти для хранения модели, где K – число деревьев.

1.2.3 Метод опорных векторов

Идея метода опорных векторов (SVM) состоит в поиске N-гиперплоскости, осуществляющей деление пространства на два подпространства, каждое из которых содержит точки только из одного кластера.

Цель SVM в том, чтобы найти оптимальную гиперплоскость, отделяющую кластеры выборки таким образом, что случаи с одной категорией целевой переменной находятся на одной стороне плоскости и случаи с другой категорией находятся на другой. Вектора, проходящие возле гиперплоскости, называются опорными векторами. На рисунке 1.4 показан результат работы SVM в двумерном пространстве.

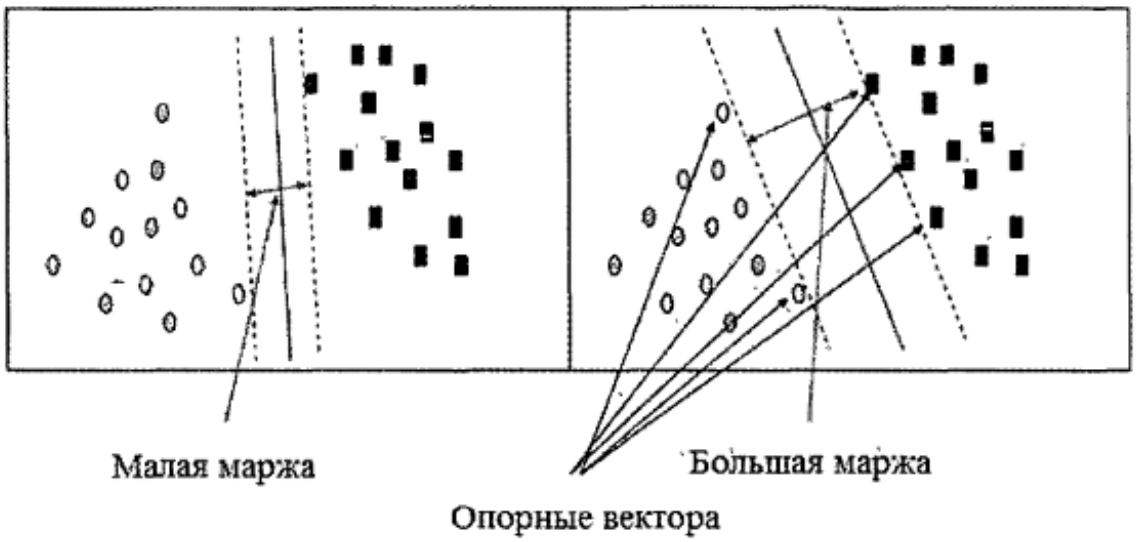

Рисунок 1.4 – Пример SVM [9]

На данном рисунке пунктирные линии отмечают расстояние между разделяющей линией и ближайшими точками. Расстояние между пунктирными линиями называется маржа. Вектора, которые ограничивают ширину маржи, называются опорными векторами.

Для определения принадлежности точки одному из кластеров строится разрешающая функция, называемая классификатором. Рассмотрим построение классификатора в простейшем случае разделения множества точек на два класса гиперплоскостью.

Обучающая выборка $(x_1, y_1), ..., (x_m, y_m), x_i \in \mathbf{R}^n, y_i \in \{-1, 1\}$. Классификатор F строится в виде функции $F(x) = sign(< w, x > +b)$, где $<,>$ – скалярное произведение, w – нормальный вектор к разделяющей гиперплоскости, b – вспомогательный параметр. Точки, для которых значение классификатора $F(x) = 1$ попадают в один класс, а объекты со значением классификатора $F(x) = -1$ – в другой класс. Функция-классификатор выбирается в таком виде исходя из того, что любая гиперплоскость определяется уравнением $< w, x > +b = 0$ для некоторых w и b.

Значения вектора w и скаляра b выбираются исходя из требования максимального расстояния до каждого класса. Расстояние определяется значением $\frac{1}{||w||}$. Таким образом, необходимо найти максимум функции $\frac{1}{||w||}$ или, что то же самое, найти минимум значения $||w||^2$. Сформулируем эти условия в виде задачи оптимизации:

$$\begin{cases} \text{argmin}||\text{w}||^2, \\ \text{y}_i(< w, x_i > +b) \geq 1, i = 1, ..., m. \end{cases} \tag{5}$$

Таким образом, построение классификатора сводится к стандартной задаче квадратичного программирования [7], которая может быть решена с помощью множителей Лагранжа [8].

Выражение $k(x, x') =< \phi(x), \phi(x') >$ принято называть ядром классификатора. В качестве ядра функции может быть выбрана любая положительно определенная симметричная функция от двух переменных. Требование положительной определенности вытекает из того, что соответствующая функция Лагранжа в задаче оптимизации должна быть ограничена снизу для корректного определения задачи оптимизации. Точность работы классификатора в большой степени определяется правильностью выбора ядра.

Преимуществами данного метода являются:

— высокая точность классификации текстовых документов в задачах распределения документов на естественном языке по заданным категориям;

— с помощью выбора ядра к методу опорных векторов может быть сведен большой класс алгоритмов искусственного интеллекта, например нейросетей;

— широкие возможности автоматизации выбора классифицирующей функции.

Недостатками же являются:

— отсутствие алгоритма для выбора ядра для произвольной задачи классификации;

— процесс обучения обладает достаточно низкой скоростью;

— необходимость работать в пространствах больших размерностей (например при классификации текстовых документов размерность пространства признаков достигает несколько десятков тысяч, в таком случае приходится прибегать к алгоритмам уменьшения размерности пространства).

1.2.4 Нейронные сети

Наиболее распространенными являются классификаторы, использующие искусственные нейронные сети [9]. Самым распространенным является использование нейросетевых классификаторов в задачах распознавания образов в графических объектах [10, 11]. Однако, алгоритмы разработанные для изображений могут быть адаптированы для текстовой информации. Классы объектов представляются в виде многомерных векторов. На основе набора обучающих векторов определяются веса синапсов искусственных нейронов. Процесс классификации при этом существенно зависит от способа представления текста в виде многомерного вектора, а также топологией нейронной сети функцией активации нейронов. Выбор алгоритма обучения классификатора определяется топологией нейронной сети.

Использование нейронных сетей в задачах классификации наталкивается на ряд трудностей. Прежде всего, размер нейросети определяется размерностью векторов, которые в ярде случаев не могут быть определены заранее. Одним из выходов является создание сетей с количеством нейронов, заведомо превышающим объем задачи. Однако это приводит к лишнему расходованию ресурсов и усложнению структуры сети.

При использовании нейросетей только с одним скрытым слоем происходит преобразование вектора входного образа в новое пространство другой размерности. Затем происходит разделение на классы. При этом происходит распознавание не только характеристик входного набора данных, но и дополнительных данных сформированных скрытым слоем.

Кроме этого для построения классификатора необходимо провести выделение признаков, существенных для классификации. Прежде всего, необходимо определить количество существенных признаков. Если признаков будет слишком мало, то возрастет процент неверно классифицированных объектов, так как отличие между классами будет слишком маленьким. При чрезмерно большом количестве признаков не только возрастает объем вычислений, но потери существенных признаков в общем объеме данных. При этом обучение нейросети сведется просто к запоминанию обучающего набора, а работа нейросети будет не корректной.

Третья проблема построения нейросетевых классификаторов возникает

при выборе способа представления входных данных, особенно их нормировка. Требование нормировки вытекает из того, что нейронные сети обрабатывают числа в промежутке от 0 до 1, тогда как диапазон входных данных определяется условиями решаемой задачи и может быть гораздо шире, а в некоторых случаях и вовсе не числовым. Самым простым методом нормировки является линейное преобразование, однако в ряде случаев необходимо производит многомерный анализ и использовать нелинейные функции, учитывающие взаимное влияние параметров входных данных друг на друга.

Архитектура нейронной сети в большинстве случаев определяется перебором нескольких конфигураций с разной как топологией, так и количеством нейронов. Основным критерием служит размер обучающего множества и эффективность работы нейросети. Наибольшее распространение получил алгоритм обучения нейросети методом обратного распределения.

В алгоритме построения классификатора на основе нейронной сети можно выделит четыре основных этапа:

1. Формирование обучающего множества.
2. Предобработка данных.
3. Построение, обучение и тестирование нейросети.
4. Использование и диагностика нейросети.

Качество построенного классификатора существенно зависит от выбора данных для обучающего и тестового множества. При отсутствии достаточно полного набора исходных данных построение эффективного классификатора невозможно.

1.3 Сравнительный анализ методов классификации

Исходя из вышеописанного разбора методов классификации, были выявлены следующие критерии оценки методов:

1. К1 – устойчивость к выбросам;
2. К2 – требование к тщательной настройке параметров;

3. К3 – возможность переобучения;

4. К4 – возможность прогнозирования непрерывных функций;

5. К5 – формальность вывода;

6. К6 – возможность работы с разреженными данными.

На таблице 1.1 изображен сравнительный анализ методов в соответствии с вышеприведенными критериями.

Таблица 1.1 – Сравнительный анализ методов классификации

	К1	К2	К3	К4	К5	К6
Логистическая регрессия	Нет	Да	Да	Нет	Да	Да
Случайный лес	Да	Нет	Да	Да	Нет	Да
Метод опорных векторов	Нет	Нет	Нет	Да	Да	Да

1.4 Формализация и постановка задачи

Требования к разрабатываемому методу:

— метод должен иметь конечное время работы;

— метод должен обрабатывать любые текстовые сообщения;

— возможность загрузки сообщений через пользовательский интерфейс.

Суть поставленной задачи – обработать полученное сообщение и отобразить результат в наглядном виде.

На рисунке 1.5 представлена формализация задачи в виде IDEF-0 диаграммы.

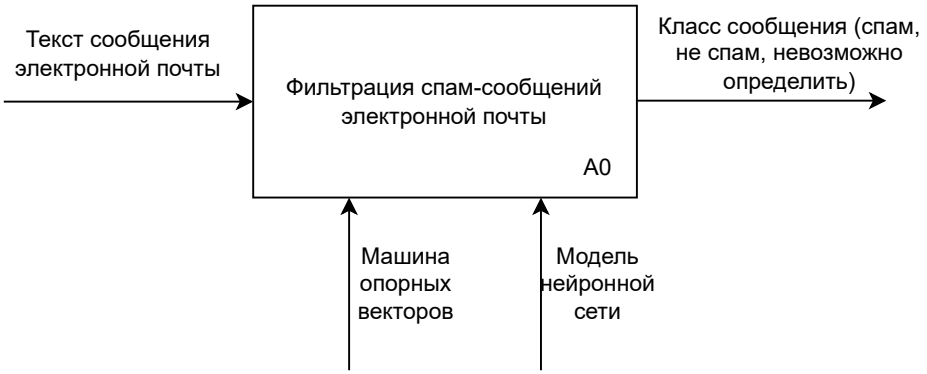

Рисунок 1.5 – IDEF0-диаграмма

Выводы из аналитического раздела

В данном разделе проведен анализ предметной области фильтрации спам-сообщений электронной почты. Приведен разбор методов классификации, а также рассмотрен анализ при помощи нейронной сети и метода опорных векторов. В качестве метода фильтрации был выбран метод на основе опорных векторов и нейронной сети.

2 Конструкторский раздел

В данном разделе сформулированы требования и ограничения к разрабатываемому методу. Описаны основные этапы разработки в виде детализированной диаграммы IDEF0 и схем алгоритмов, а также изложены особенности излагаемого метода. Спроектировано программное обеспечение для реализации разрабатываемого метода.

2.1 Требования и ограничения к разрабатываемому методу

К методу фильтрации спам-сообщений электронной почты на основе метода опорных векторов и нейронной сети предъявляются следующие требования:

— метод должен иметь конечное время работы;

— метод должен обрабатывать любые текстовые сообщения.

К разрабатываемому программному обеспечению предъявляются следующие требования:

1. Возможность загрузки текстовых сообщений через пользовательский интерфейс, в том числе txt файлы.
2. Если время выполнения программы может превышать комфортное время реакции системы для человека, необходимо обеспечить вывод предупреждающего сообщения.
3. Разрабатываемое ПО должно корректно реагировать на любые действия пользователя.

2.2 Основные этапы разрабатываемого метода

На рисунке 2.1 представлена диаграмма IDEF0 первого уровня для разрабатываемого метода.

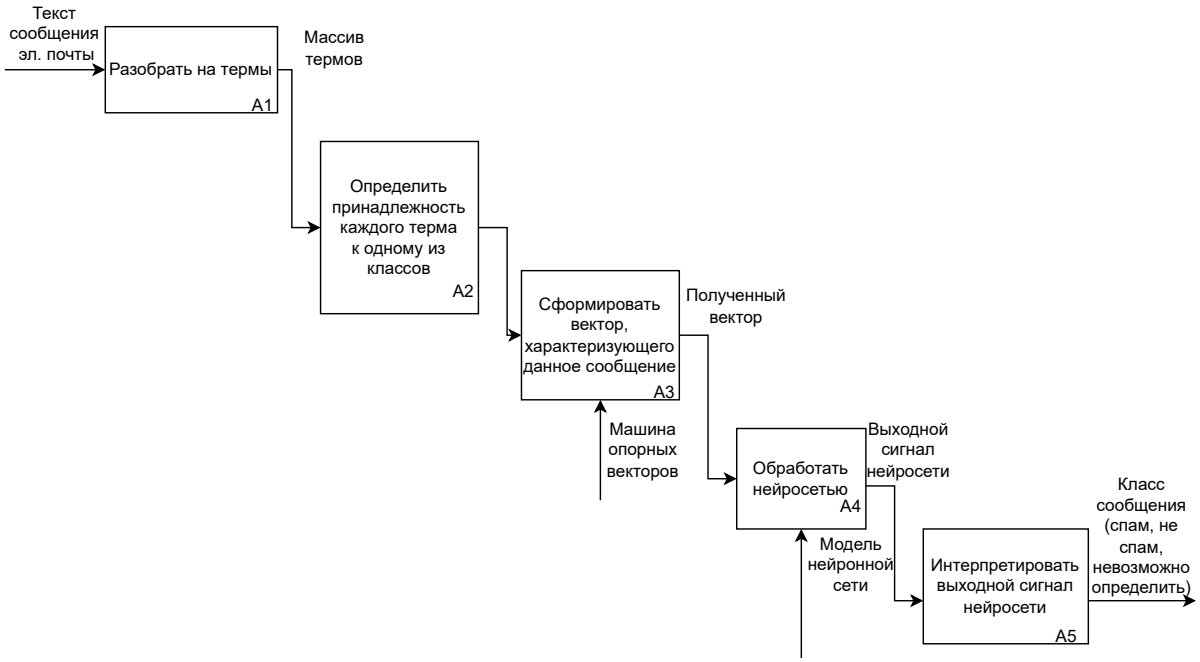

Рисунок 2.1 – IDEF0 – диаграмма первого уровня

Здесь под термом понимается слово в его начальной форме. Вектор, полученный от машины опорных векторов является вектором коэффициентов спамности соответствующих слов, находящихся в том же порядке, что и в классифицируемом сообщении.

Также, далее:

— @ – функция умножения матриц;

— {} – функция создания пустого словаря.

2.2.1 Схемы алгоритмов обучения

На первом этапе формируются данные для последующего обучения. Схема алгоритма представлена на рисунках 2.2 - 2.4.

Рисунок 2.2 – Формирование словаря коэффициентов спамности слов, ч.1

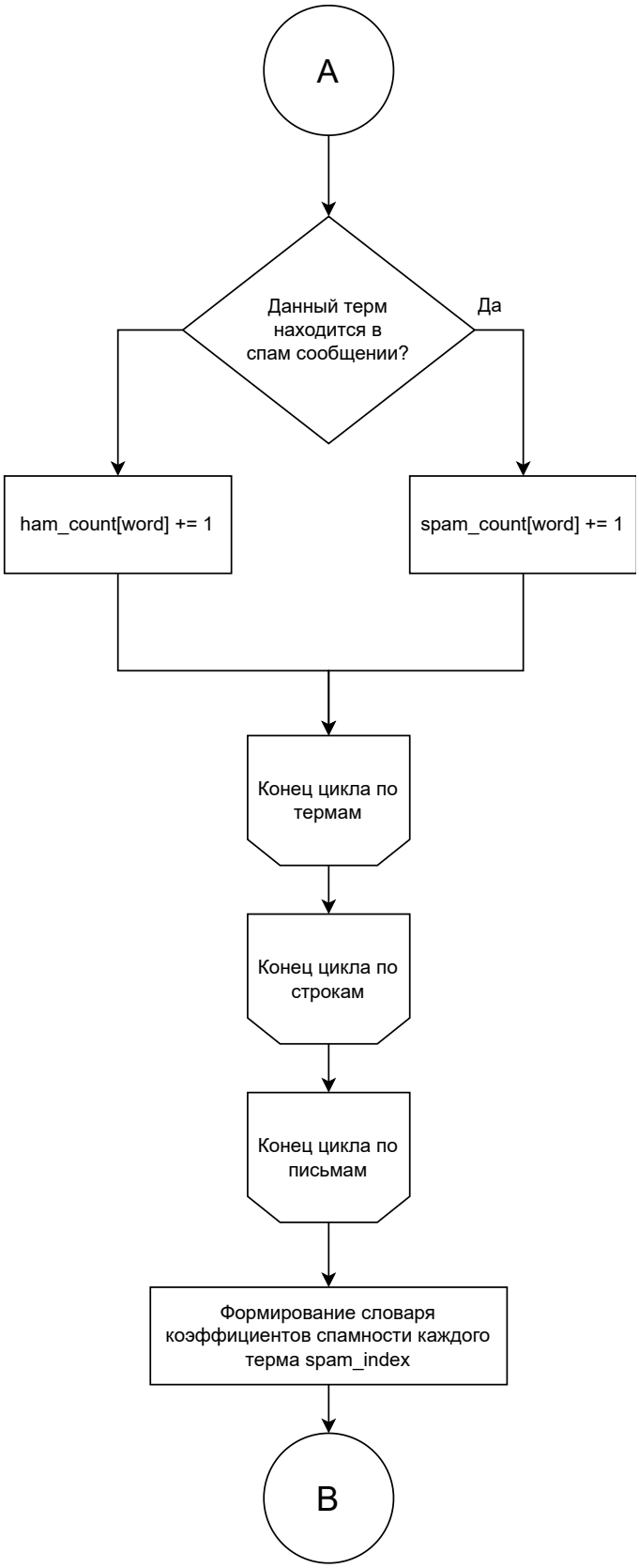

Рисунок 2.3 – Формирование словаря коэффициентов спамности слов, ч.2

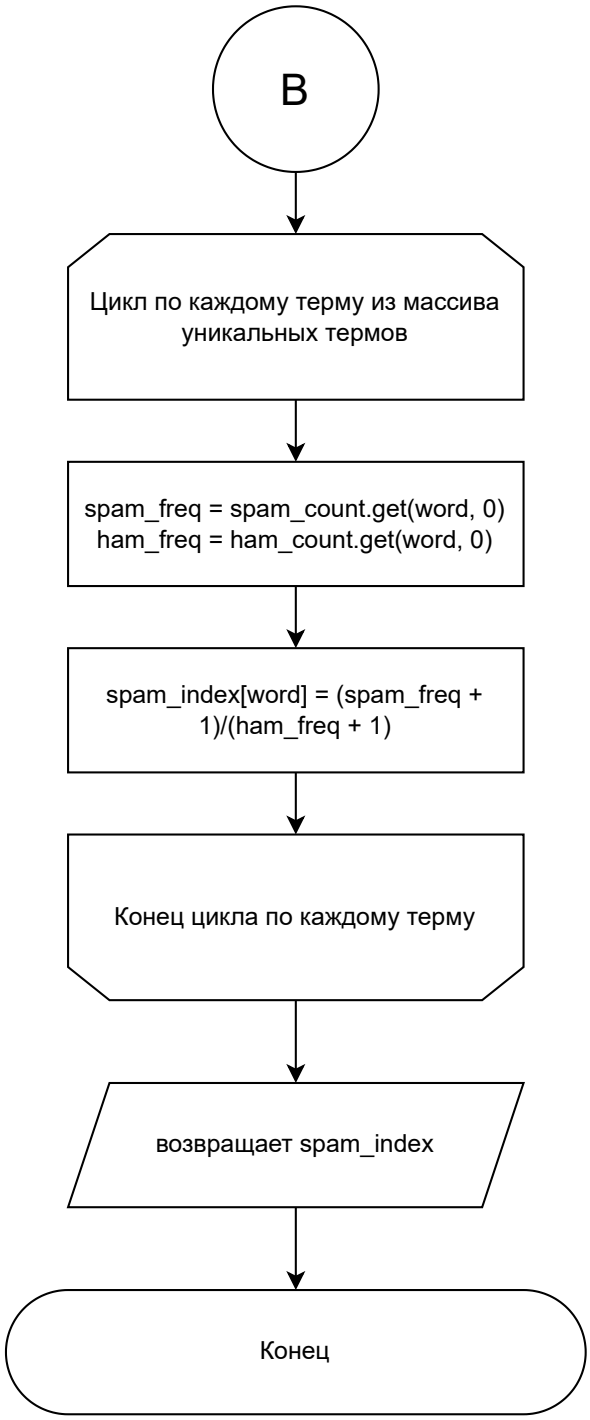

Рисунок 2.4 – Формирование словаря коэффициентов спамности слов, ч.3

В свою очередь, схема алгоритма создания матрицы признаков представлена на рисунках 2.5 - 2.6.

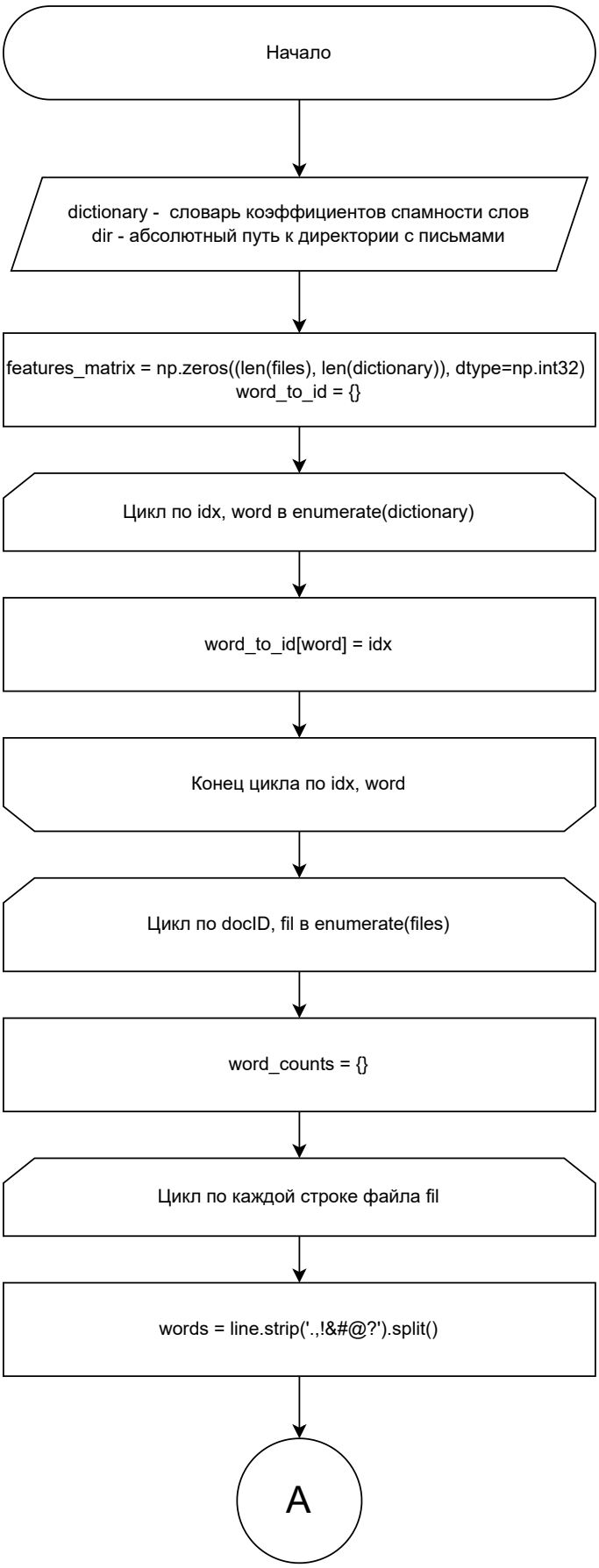

Рисунок 2.5 – Создание матрицы признаков ч.1

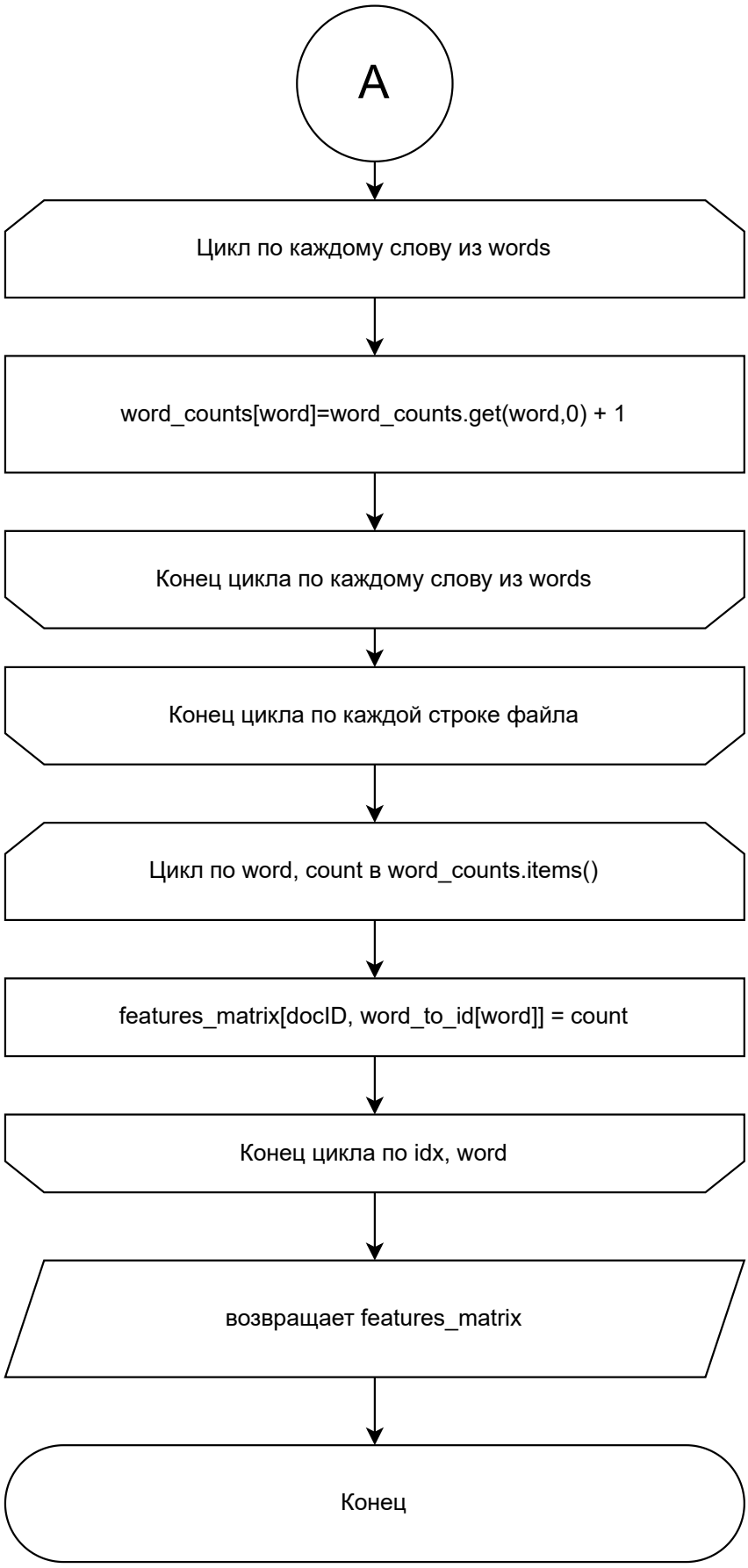

Рисунок 2.6 – Создание матрицы признаков ч.2

Обучение машины опорных векторов представлено на рисунке 2.7.

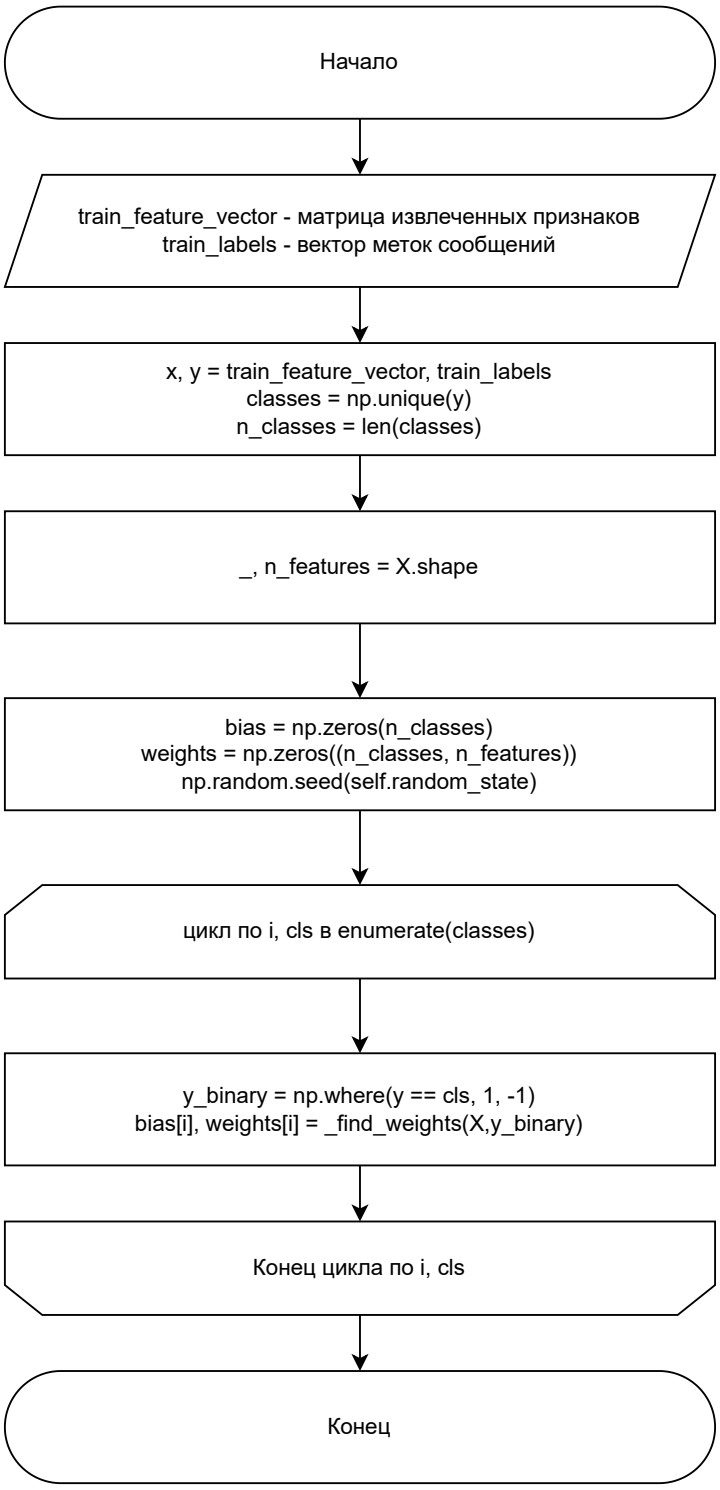

Рисунок 2.7 – Обучение машины опорных векторов

Функция find_weights, используемая при обучении машины опорных векторов, представлена на рисунке 2.8.

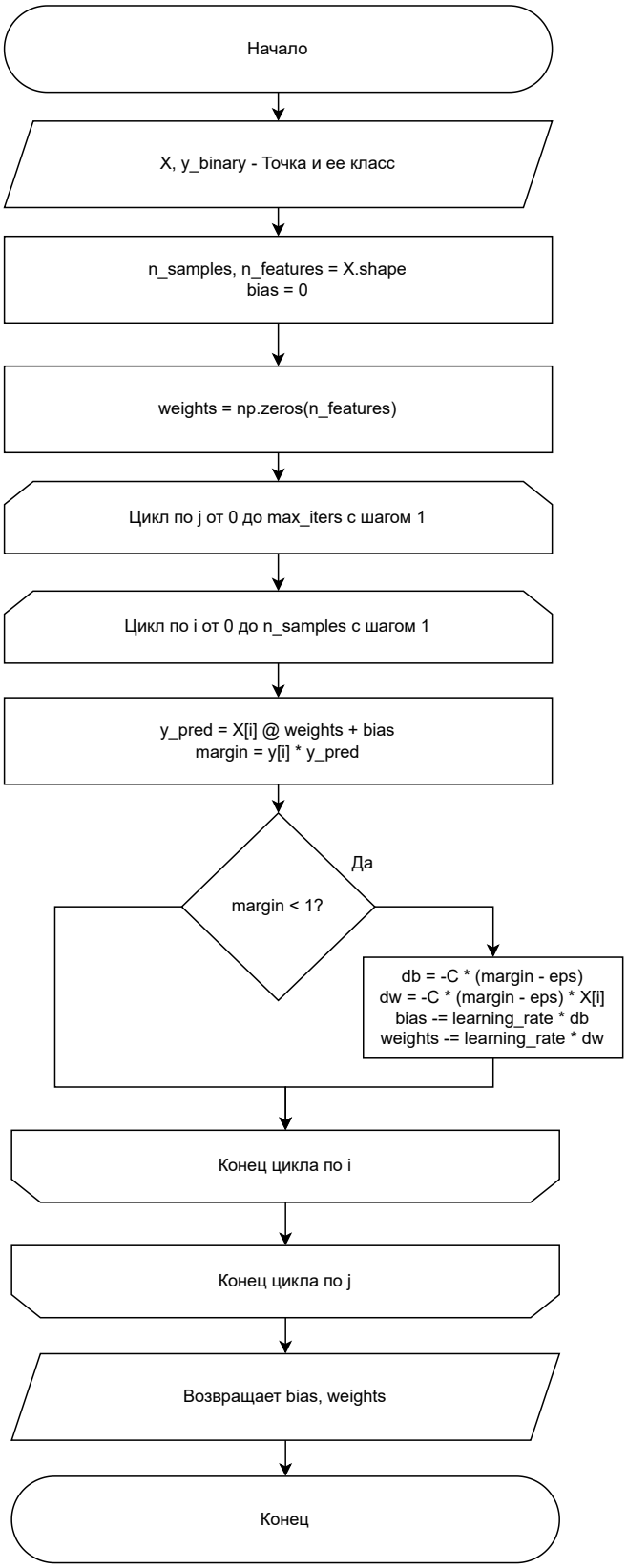

Рисунок 2.8 – Функция нахождения весов в машине опорных векторов

Само обучение нейросети представлено на рисунке 2.9.

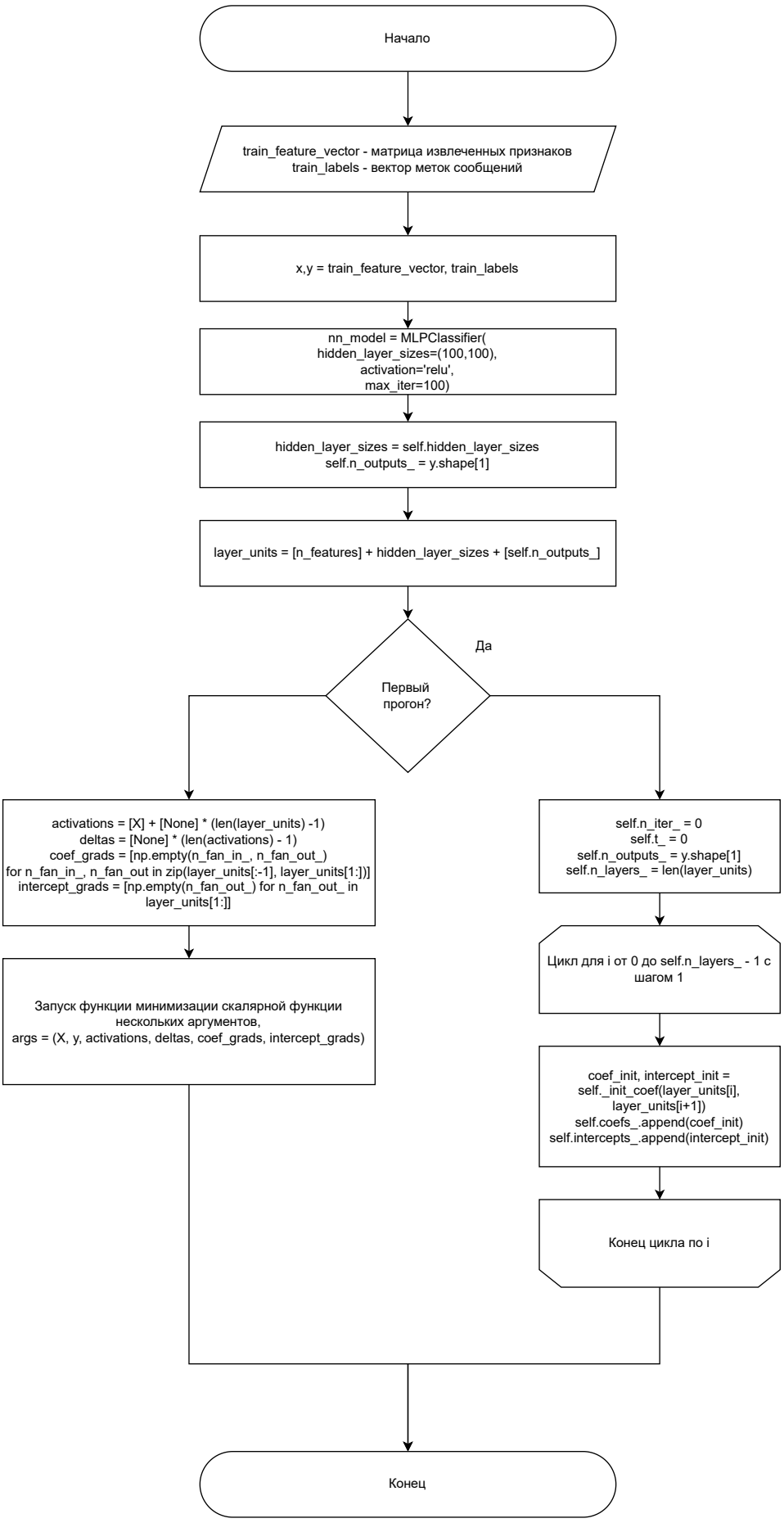

Рисунок 2.9 – Обучение нейронной сети

2.2.2 Схемы алгоритмов классификации

Схема создания вектора коэффициентов спамности представлена на рисунке 2.10.

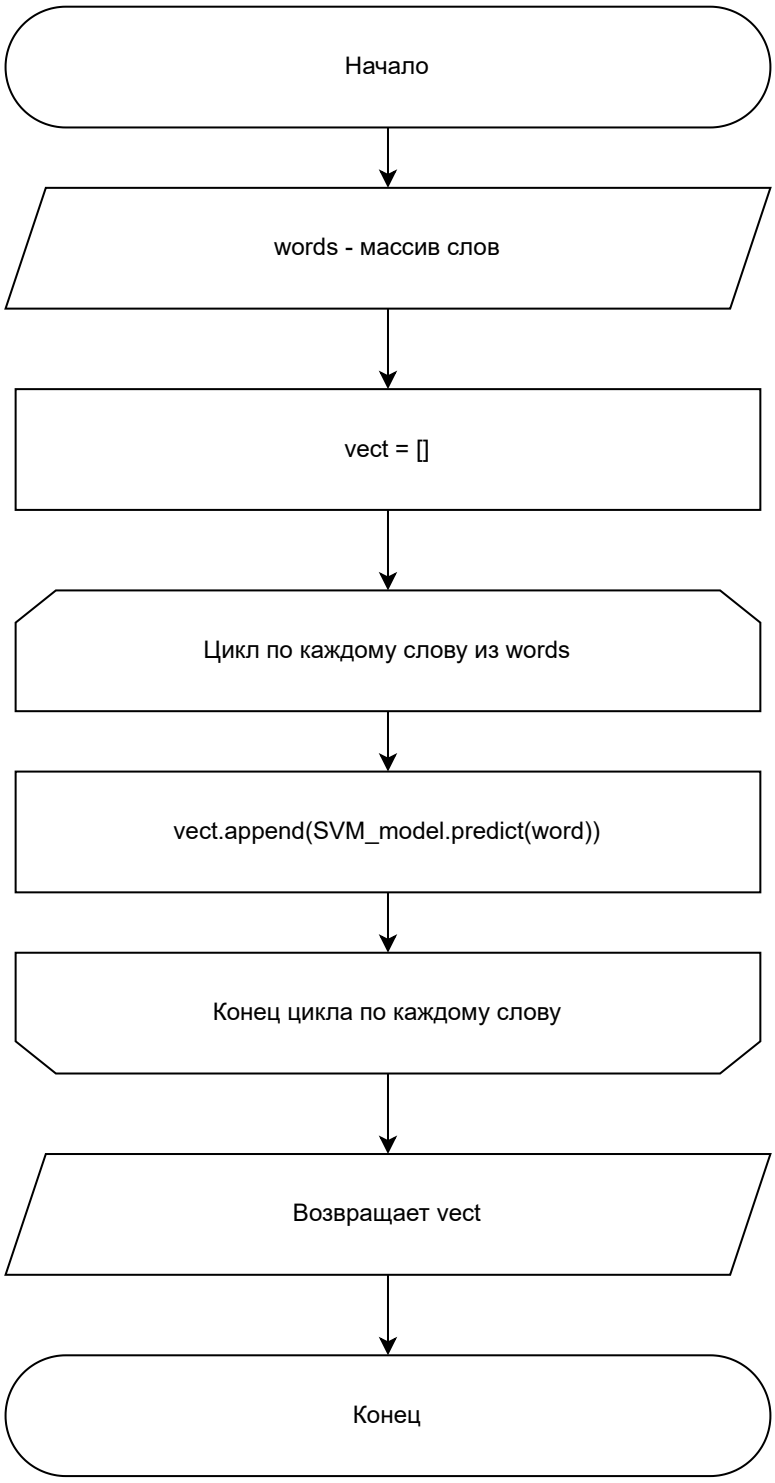

Рисунок 2.10 – Построение вектора на вход нейронной сети

Функция predict представлена на рисунке 2.11.

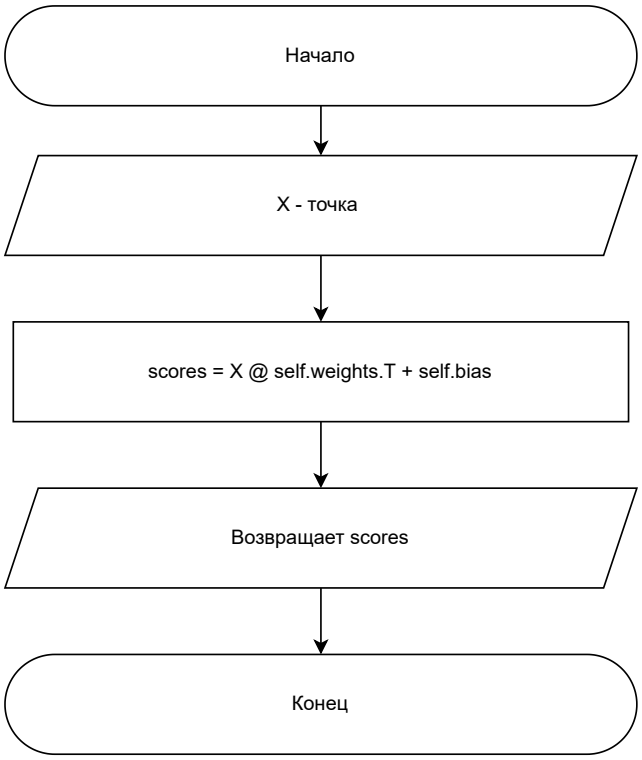

Рисунок 2.11 – Функция predict SVM-машины

На рисунке 2.12 представлен процесс обработки нейросетью полученного вектора.

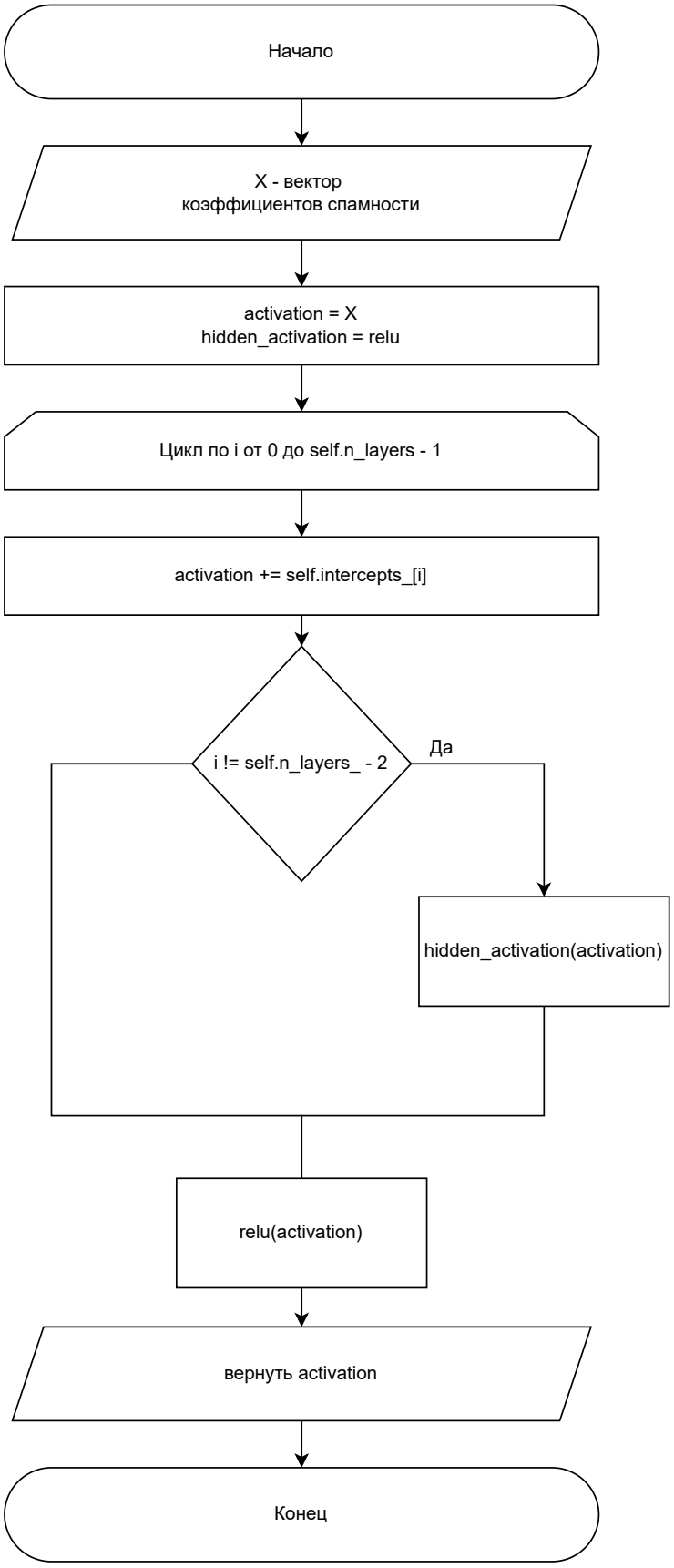

Рисунок 2.12 – Обработка нейросетью полученного вектора

Если на выходе нейросети будет получено значение выше 0.85, сообщение будет классифицировано как спам, если ниже 0.85 – не спам, иначе – невозможно определить

Выводы

В данном разделе были сформулированы требования и ограничения к разрабатываемому методу и соответствующему ПО, рассмотрены основные этапы разрабатываемого метода в виде детализированной диаграммы IDEF0 и схем алгоритмов.

3 Технологический раздел

В данном разделе обоснован выбор средств программной реализации предлагаемого метода, описан формат входных и выходных данных. Представлены детали реализации метода, а также описано взаимодействия пользователя с программным обеспечением.

3.1 Выбор средств реализации программного обеспечения

В качестве языка программирования для реализации предлагаемого метода был выбран Python [12]. Выбор обусловлен тем, что данный язык имеет большое количество библиотек для работы с нейронными сетями и методом опорных векторов.

В качестве среды разработки был выбрана Visual Studio Code [13]. Данная IDE имеет открытый исходный код, легко настраиваема и имеет огромное количество расширений для упрощения работы.

3.2 Формат входных и выходных данных

Входные данные для разрабатываемого метода:

— текст сообщения электронной почты в файле формата txt.

Выходные данные для разрабатываемого метода:

— класс отфильтрованного сообщения (спам, легитимное, невозможно определить).

3.3 Детали реализации предлагаемого метода

В листинге 3.1 представлена функция создания словаря слов с соответствующими вероятностями нахождения в легитимных и спам сообщениях.

Листинг 3.1 – Функция создания словаря слов

```
def make_spam_index(train_dir):
    emails = [os.path.join(train_dir, f) for f in
       os.listdir(train_dir)]
    all_words = []
    spam_count = {}
    ham_count = {}

    for mail in emails:
        with open(mail) as m:
            for i, line in enumerate(m):
                if i == 2:
                    words = line.split()
                    for word in words:
                        all_words.append(word)
                        if 'spam' in mail:
                            spam_count[word] = spam_count.get(word,
                               0) + 1
                        else:
                            ham_count[word] = ham_count.get(word,
                               0) + 1

    spam_index = {}
    for word in set(all_words):
        spam_freq = spam_count.get(word, 0)
        ham_freq = ham_count.get(word, 0)
        if spam_freq + ham_freq > 0:
            spam_index[word] = np.log((spam_freq + 1) / (ham_freq +
               1))

    return spam_index
```

В качестве датасета были использованы данные Ling-spam corpus, 702 сообщений из которого были выделены на обучающее множество, а 206 на тестирующее. Процентное соотношение легитимных/спам сообщений соответствует полученной раннее статистике [1].

В листинге 3.2 представлено построение вектора для последующей обработки нейронной сетью.

Листинг 3.2 – Построение вектора для последующей обработки

```
def extract_weighted_features(mail_dir, dictionary):
    files = [os.path.join(mail_dir, fi) for fi in
       os.listdir(mail_dir)]
    features_matrix = np.zeros((len(files), len(dictionary)),
       dtype=np.int32)
    word_to_id = {}
    for idx, word in enumerate(dictionary):
        word_to_id[word] = idx

    for docID, fil in enumerate(files):
        word_counts = {}
        with open(fil) as fi:
            for i, line in enumerate(fi):
                if i == 2:
                    words = line.strip().split()
                    for word in words:
                        word_counts[word] = word_counts.get(word,
                           0) + 1

        for word, count in word_counts.items():
            if word in word_to_id:
                features_matrix[docID, word_to_id[word]] = count

    return features_matrix
```

В листинге 3.3 представлено обучение машины опорных векторов и модели нейронной сети. Модели были взяты из библиотеки Sklearn, в качестве модели нейронной сети был выбран многослойный перцептрон с 2 скрытыми слоями по 100 нейронов каждый, а для SVM-машины – линейное ядро.

Листинг 3.3 – Обучение SVM-машины

```
spam_index = make_spam_index(train_dir)
train_feature_vector = extract_weighted_features(train_dir,
   spam_index)
test_feature_vector = extract_weighted_features(test_dir,
   spam_index)
```

Листинг 3.4 – Обучение нейронной сети

```
nn_model = MLPClassifier(
    hidden_layer_sizes=(100, 100),
    activation='tanh',
    max_iter=50000,
    alpha=0.1,
    batch_size='auto',
    learning_rate_init=0.001,
    learning_rate='adaptive',
    early_stopping=True,
    validation_fraction=0.2,
    n_iter_no_change=10
)

nn_model.fit(train_feature_vector, train_labels)
```

3.4 Описание взаимодействия пользователя с программным обеспечением

На рисунке 3.1 представлен разработанный пользовательский интерфейс программного обеспечения.

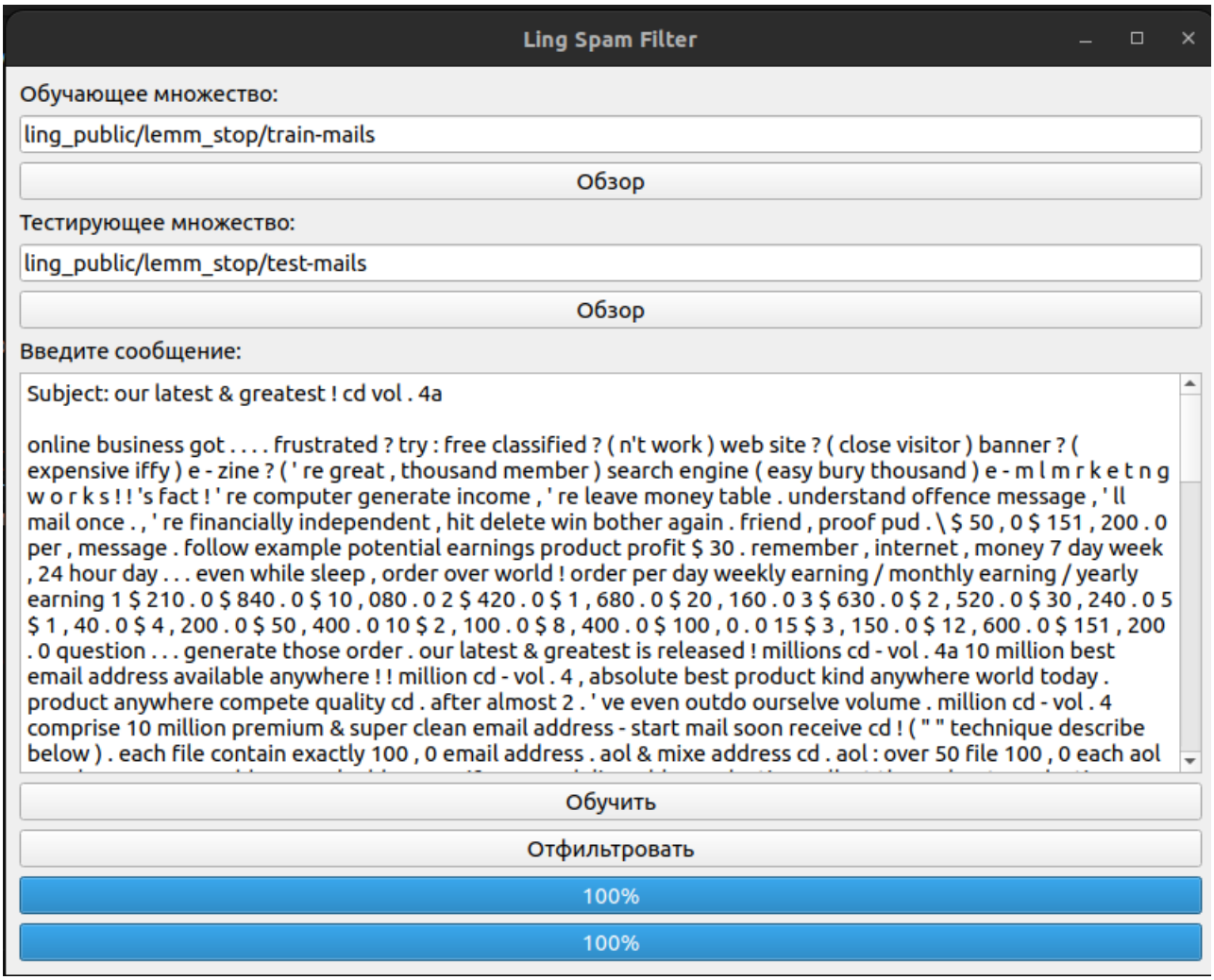

Рисунок 3.1 – Пользовательский интерфейс программного обеспечения

Разработанный интерфейс позволяет выбрать тестирующее и обучающее множества для нейронной сети и машины опорных векторов. Процесс обучения и тестирования может превышать комфортное время реакции интерактивной системы для человека, в связи с чем в интерфейсе существует две полоски прогресса, первая для обучения, вторая – для тестирования. Также в результате работы программы выводится таблица с результатами тестирования, где можно посмотреть, сколько соответствующих классов сообщений было отфильтровано.

При попытке запуска программы без предварительного выбора множеств выводится специальное уведомление для пользователя (рисунок 3.2).

При попытке запуска фильтрации без предварительного обучения моделей также выводится уведомление.

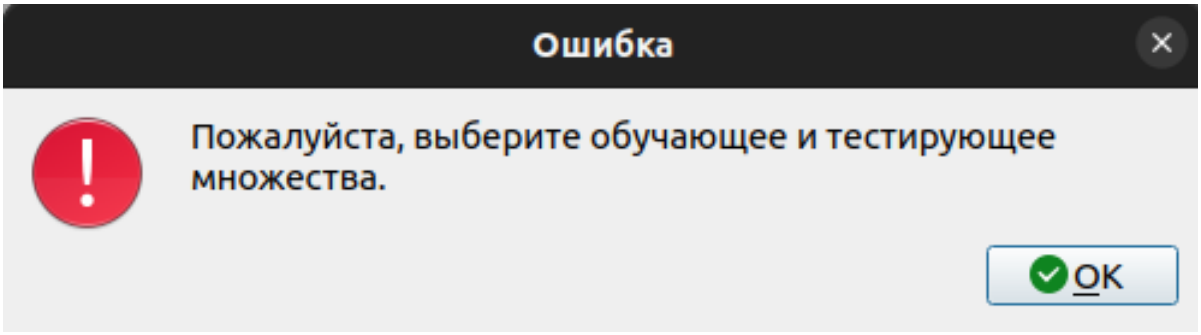

Рисунок 3.2 – Уведомление о необходимости предварительной загрузки изображения

3.5 Сборка и запуск проекта

В случае, если на персональный компьютер пользователя уже установлен Python версии 3.10.2, то для сборки и запуска разработанного ПО необходимо выполнить следующие шаги:

— проверить наличие библиотек PyQT5 и Scikit-learn;

— проверить, что все необходимые файлы и множества находятся в той же папке, что и основной файл программы;

— открыть папку проекта;

— запустить ПО либо через консольный интерфейс.

В случае, если Python 3.10.2 или необходимые библиотеки не установлены, то необходимо выполнить следующие шаги:

— проверить, что исполняемый файл и другие файлы, связанные с проектом, находятся на целевой машине пользователя;

— загрузить с официального сайта Python последнюю версию интерпретатора;

— пройти инсталляцию необходимых библиотек, в данном случае потребуется ввести в консоли команды, приведенные на листинге 3.5;

Листинг 3.5 – Установка требуемых компонентов программы

```
pip install pyqt5
pip install scikit-learn
```

— запустить проект.

4 Исследовательский раздел

В данном разделе приведены замеры точности и времени работы реализованного метода. Проведено сравнение результатов работы метода с результатами, полученными от известных аналогов.

4.1 Технические характеристики

Технические характеристики машины, на которой производились исследования:

— операционная система: Linux Ubuntu 22.04 64-bit [14];

— оперативная память: 16 Gb;

— процессор: 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz [15].

4.2 Исследование времени и точности работы предложенного метода

Постановка исследования

Исследование заключается в определении времени обучения машины опорных векторов и нейронной сети в зависимости от размера обучающей выборки, а также точности работы разработанного метода. Также проводится исследование времени работы метода в зависимости от размера тестирующего множества.

Для замеров было использовано по 4 обучающих и тестирующих множеств. Замеры проводились несколько раз для минимизации влияния случайных факторов.

Результаты исследования

В таблице 4.1 представлены результаты замеров времени обучения SVM и нейронной сети в зависимости от размера обучающей выборки. Размер выборки измеряется в количестве сообщений, а время — в секундах. Для замеров времени выполнения использовались команды *time()*, представленные в библиотеке *time* для Python.

Таблица 4.1 – Зависимость времени обучения машины опорных векторов и нейронной сети от размера обучающей выборки

Количество сообщений в выборке	SVM, сек.	Нейросеть, сек.
130	0.12	2.5
260	0.14	3.98
450	0.17	5.12
702	0.2	10.23

В таблице 4.2 представлена точность метода в зависимости от размера обучающей выборки. Точность вычислялась как соотношение корректно распознанных сообщений на число всех сообщений в выборке.

Таблица 4.2 – Зависимость точности разработанного метода от размера обучающей выборки

Количество сообщений в выборке	Точность, %
130	50
260	52
450	73
702	95

На рисунке 4.1 представлена зависимость времени обучения SVM и нейросети количества сообщений в обучающей выборке.

На рисунке 4.2 представлена зависимость точности разработанного метода от количества сообщений в обучающей выборке.

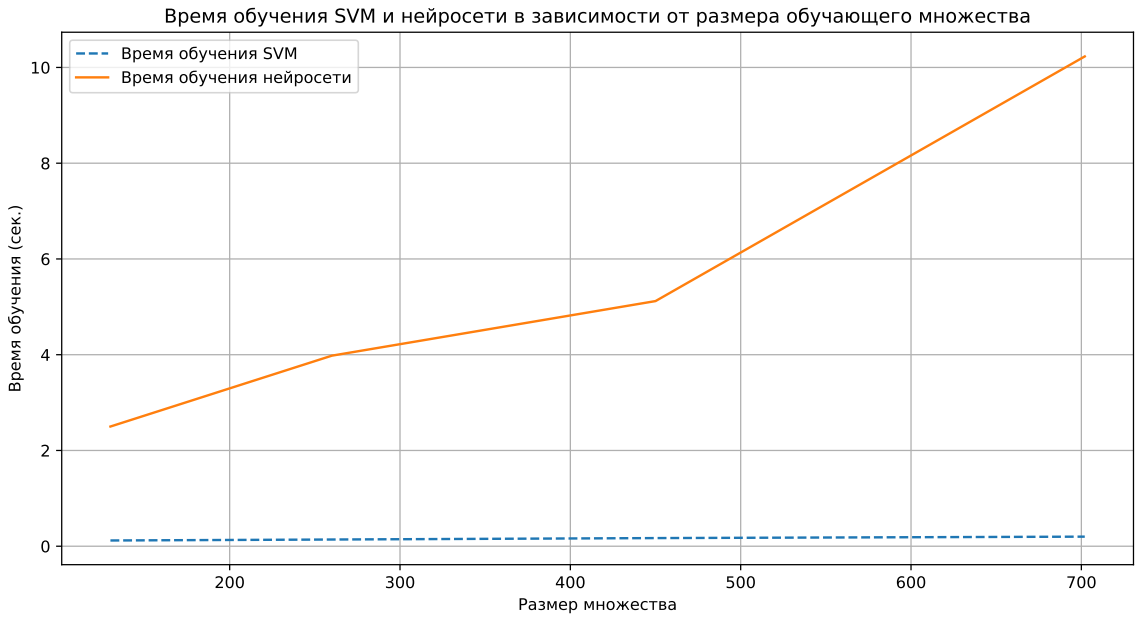

Рисунок 4.1 – Зависимость времени обучения SVM и нейросети количества сообщений в обучающей выборке

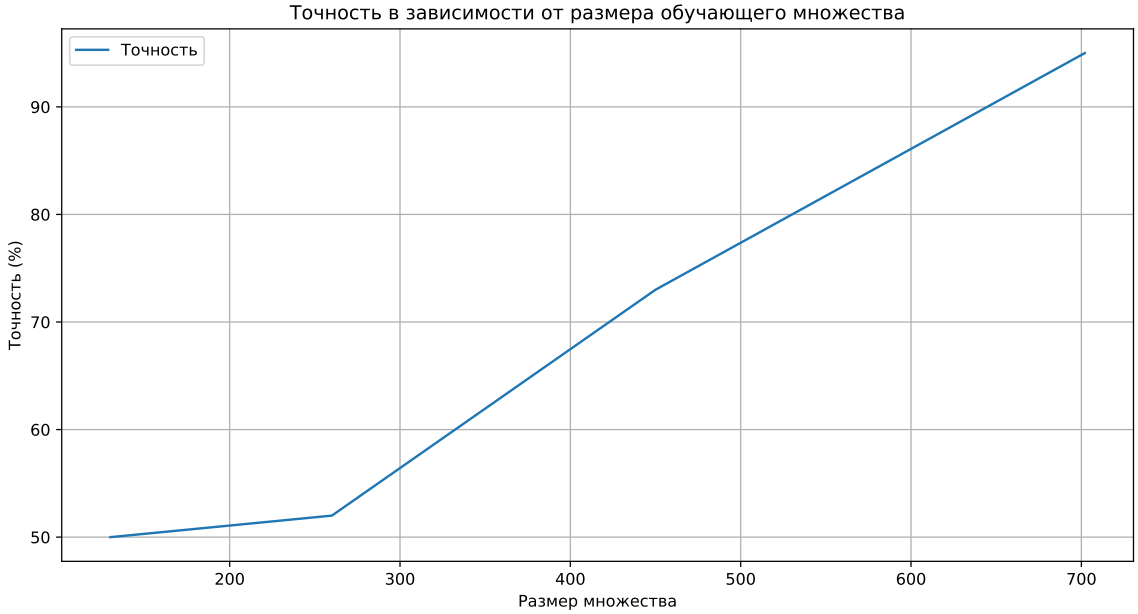

Рисунок 4.2 – Зависимость точности разработанного метода от количества сообщений в обучающей выборке

4.3 Сравнение с результатами, полученными от известных аналогов

В данном разделе будет произведено сравнение точности разработанного в этой работе метода и известных аналогов. Также, в качестве исходных данных будет использована статистика, полученная от AV-Comparatives, Virus Bulletin и PCMag за 2022 и 2023 ввиду того, что данные о точности и скорости спам-фильтров от таких компаний как Microsoft или Google являются коммерческой тайной. Также в сравнении будут представлены основные методы работы данных сервисов.

В таблице 4.3 представлено сравнение точности работы разработанного метода и известных аналогов. Сравнение проводилось по следующим критериям:

— Точность.

— К1 – Использование правил для фильтрации.

— К2 – Использование сигнатурного анализа.

— К3 – Использование репутационных методов.

— К4 – Использование машинного обучения.

Таблица 4.3 – Сравнение точности работы метода и его аналогов

Спам-фильтр	Точность	К1	К2	К3	К4
Разработанный метод	95%	Нет	Нет	Нет	Да
Microsoft 365 Defender	97%	Да	Нет	Да	Да
Proofpoint	95%	Да	Нет	Да	Да
Fortinet FortiMail	96%	Да	Да	Да	Да
Trend Micro	98%	Да	Да	Да	Да
Symantec	96%	Да	Да	Нет	Да
Sophos	95%	Да	Да	Нет	Да
Barracuda Essentials	94%	Да	Нет	Да	Да
Cisco Secure Email	93%	Да	Нет	Да	Да
Zoho Mail	92%	Да	Нет	Да	Да

Выводы

Было проведено исследование разработанного метода и сравнение данного метода с известными аналогами. По результатам исследования выяснилось, что точность разработанного метода при выборке в 702 сообщения достигает 95%, что позволяет конкурировать с существующими аналогами. Также, данная точность была достигнута только за счет использования методов машинного обучения.

ЗАКЛЮЧЕНИЕ

В данной работе была рассмотрена фильтрации спам-сообщений электронной почты на основе метода опорных векторов и нейронной сети.

Для достижения поставленной задачи был проведен анализ предметной области спам сообщений, а также существующих методов фильтрации спама.

Для решения задачи был предложен метод, использующий только машинное обучение.

В результате выполнения данной работы было спроектировано и реализовано соответствующее программное обеспечение, позволяющее пользователю загружать сообщения электронной почты для проверки на спам.

Разработанный метод может конкурировать с известными аналогами, предоставляя приблизительно схожую точность.

В качестве перспектив дальнейшего развития можно рассмотреть несколько направлений, таких как добавление новых методов для анализа сообщений; интеграцию в почтовые сервисы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Securelist [Электронный ресурс]. Режим доступа, URL: https://securelist.ru/spam-phishing-report-2023/109104/ (дата обращения 29.04.2024).

2. Картаслов [Электронный ресурс]. Режим доступа, URL: `https://kartaslov.ru/znachenie-slova/spam` (дата обращения 29.04.2024).

3. Kasperskiy [Электронный ресурс]. Режим доступа, URL: `https://www.kaspersky.ru/about/press-releases/2022_dva-dnya-v-god-stolko-vremeni-ofisnye-sotrudniki-tratyat` (дата обращения 29.04.2024).

4. Almeida M. B., Braga A. P., Braga J. P.: SVM-KM: speeding SVMs learning with a priori cluster selection and k-means. In: Proceedings of the 6th Brazilian Symposium on Neural Networks, 2000. p. 162-167

5. LearnStatistics [Электронный ресурс]. Режим доступа, URL: `https://ru.statisticseasily.com/logistic-regression-scikit/` (дата обращения 29.04.2024).

6. LearnStatistics [Электронный ресурс]. Режим доступа, URL: `https://progler.ru/blog/preimuschestva-i-nedostatki-logisticheskoy-re` (дата обращения 29.04.2023).

7. LearnStatistics [Электронный ресурс]. Режим доступа, URL: `https://ru.statisticseasily.com/random-forest/` (дата обращения 29.04.2024).

8. Акулич И.Л. Глава 3. Задачи нелинейного программирования // Математическое программирование в примерах и задачах— М.: Высшая школа, 1986. — С . 319.

9. Бугров Я. С, Никольский С. М. Высшая математика. Дифференциальные уравнения. Кратные интегралы. Ряды. Функции комплексного переменного — М.: Наука, 1985. — С. 464.

10. Головко В.А. Нейронные сети: обучение, организация и применение. М., ИПРЖР, 2001.

11. Burghouts G.J., Geusebroek J.M. Performance evaluation of local colour invariants // Computer Vision and Image Understanding. - 2009. - V. 113, - P . 48-62.

12. Python [Электронный ресурс]. Режим доступа, URL: `https://www.python.org/` (дата обращения 30.04.2024).

13. VSCode [Электронный ресурс]. Режим доступа, URL: `https://code.visualstudio.com/` (дата обращения 30.04.2024).

14. Ubuntu [Электронный ресурс]. Режим доступа, URL: `https://ubuntu.com/desktop` (дата обращения 30.04.2024).

15. Intel [Электронный ресурс]. Режим доступа, URL: `https://www.intel.com` (дата обращения 30.04.2024).

ПРИЛОЖЕНИЕ А
Презентация работы

Презентация выпускной квалификационной работы содержит 19 слайдов, на которых представлено краткое описание выпускной квалификационной работы.