DOI: https://doi.org/10.36910/6775-2524-0560-2021-42-03
УДК 004.4'24
Безкостна Катерина Петрівна, студентка
Гришанович Тетяна Олександрівна, канд. фіз.-мат. наук, старший викладач
https://orcid.org/0000-0002-3595-6964
Глинчук Людмила Ярославівна, канд. фіз.-мат. наук, доцент
https://orcid.org/0000-0002-8943-9604
Булатецький Віталій Вікторович, канд. фіз.-мат. наук, доцент
https://orcid.org/0000-0002-9883-4550
Волинський національний університет імені Лесі Українки, м. Луцьк, Україна

РОЗРОБКА РОЗШИРЕННЯ ДО БРАУЗЕРА GOOGLE CHROME ДЛЯ БЛОКУВАННЯ ГРАФІЧНОГО КОНТЕНТУ

Безкостна К. П., Гришанович Т. О., Глинчук Л. Я., Булатецький В. В. Розробка розширення до браузера Google Chrome для блокування графічного контенту. У роботі розглянуто питання, що стосуються процесу розробки розширення до браузера, яке блокуватиме контент за певною ознакою. Для блокування графічного контенту використано бібліотеку NSFW.js, яка у свою чергу використовує відкриту програмну бібліотеку для машинного навчання TensorFlow. Розроблено розширення для браузера Google Chrome, що блокує графічні зображення, які потрапляють під категорію 18+. При розробці використані HTML, CSS, JavaScript.
Ключові слова: NSFW, TensorFlow, Google Chrome, розширення для браузера, фреймворк, машинне навчання, графічні зображення, растрові зображення.

Безкостная К. П., Гришанович Т. А. , Глинчук Л. Я., Булатецкий В. В. Разработка расширения к браузеру Google Chrome для блокировки графического контента. В работе рассмотрены вопросы, касающиеся процесса разработки расширение для браузера, которое будет блокировать контент по определенному признаку. Для блокировки графического контента использовано библиотеку NSFW.js, которая в свою очередь использует открытую программную библиотеку для машинного обучения TensorFlow. Разработано расширения для браузера Google Chrome, что блокирует графические изображения, попадающие под категорию 18+. При разработке использованы HTML, CSS, JavaScript.
Ключевые слова: NSFW, TensorFlow, Google Chrome, расширение для браузера, фреймворк, машинное обучение, графические изображения, растровые изображения.

Beskostna K. P., Hryshanovych T. O., Hlynchuck L. Y., Bulatetsky V. V. Development of Google Chrome browser extension for blocking graphic content. The paper considers the main issues related to the process of developing a browser extension for blocking content on a certain basis. To block graphic content the NSFW.js library was used. NSFW.js uses the open software library for machine learning TensorFlow. Google Chrome browser extension for blocking content that fall into the 18+ category was developed t. HTML, CSS, JavaScript were used in the extension development.
Ключевые слова: NSFW, TensorFlow, Google Chrome, browser extensions, framework, machine learning, images, raster images.

Постановка проблеми та аналіз досліджень. NSFW (not safe for work, not suitable for work) – це термін, який застосовується до контенту (веб-сайти, вкладення електронної пошти), що є неприйнятним для перегляду на робочому місці. [1] Даний контент не обов'язково блокується. Якщо такі соціальні мережі, як Instagram, Facebook і Tumblr жорстко цензурують своїх користувачів, Reddit дозволяє опційно вмикати/вимикати режим NSFW, то, наприклад, Twitter має анонімні аккаунти, що публікують заборонені матеріали. Крім того, соціальні мережі є не єдиним способом поширення контенту, що вважається забороненим, для цього використовуються і веб-сайти. Якщо мова йде про обмеження контенту для неповнолітніх, то для цього можуть бути використані, або вбудовані можливості операційної системи, або додаткові сервіси. Наприклад, значного поширення наразі набув засіб FamilyLink від Google [2]. Але він орієнтований на операційну систему Android та не дозволяє переглядати історію пошуку браузера. З цієї причини постає потреба у розробці програмного засобу, що дозволяв би не лише приховати зображення, але й зберігав би їх у історії пошуку. Найефективнішим вирішенням цієї проблеми може бути розробка розширення для браузера, оскільки такі програмні засоби не вимагають встановлення, інформують (за допомогою графічних елементів на панелі навігації) користувача, що його контент піддається обмеженню. Якщо розглядати найпопулярніший браузер – Google Chrome (за даними статистики сайту StatCounter [3], браузер Chrome є абсолютним лідером за популярністю у світі, а його ринкова частка у світі серед найпопулярніших веб-оглядачів на всіх платформах у лютому 2021 склала 63,63%), то розробка розширення для нього є обґрунтованою. Кожна вкладка у ньому є окремим процесом і розширення можна застосовувати для кожної вкладки окремо.

© Безкостна К. П., Гришанович Т. О., Глинчук Л. Я., Булатецький В. В.

Метою роботи є дослідження механізмів блокування зображень на основі заданих характеристик та розробка розширення для браузера Google Chrome, який блокуватиме графічний контент, що відноситься до категорії 18+ або до категорії забороненого чи неприйнятного. При цьому неприйнятні зображення не повинні видаленими із історії пошуку. Розширення для браузера повинно знаходити всі зображення на веб-сторінці, перехоплювати їх та блокувати до завершення перевірки на заборонений контент, після чого показувати лише дозволені зображення. З метою інформування користувача про роботу розширення браузера його логотип повинен відображатись на панелі додатків та розширень, а блокування забороненого зображення повинно відбуватись за допомогою відповідного стилю в CSS, що виконує розмиття по Гаусу. Також користувач повинен мати можливість увімкнути та вимкнути розширення. Така програмна розробка може бути використана для обмеження перегляду графічного контенту за допомогою браузера на персональних комп'ютерах, які використовуються в офісних приміщеннях, у комп'ютерних класах навчальних закладів та закладів позашкільної освіти, а також на домашніх персональних комп'ютерах.

Виклад основного матеріалу й обґрунтування отриманих результатів дослідження.

Для написання розширення до браузера Google Chrome для блокування неприйнятного графічного контенту, використовували мову програмування JavaScript (JS), мову розмітки гіпертексту – HTML та таблицю каскадних стилів – CSS. В якості середовища розробки було вибрано комерційне крос-платформне інтегроване середовище JetBrains PhpStorm. PhpStorm розроблений на основі платформи IntelliJ IDEA, написаної на Java. [4] Крім того, користувачі мають можливість розширити функціональність середовища розробки за рахунок установки плагінів, розроблених для платформи IntelliJ, або написавши власні плагіни.

Головним файлом для розширення є «manifest», в якому прописується версія, назва, взаємозв'язок файлів, різного роду налаштування, тощо.

Використовуючи manifest.json, визначаються основні метадані про розширення, такі як ім'я та версія. Також можна визначити деякі аспекти функціональності (такі, як фонові скрипти, постійні скрипти та дії браузера).

Маніфести веб-додатків є частиною колекції веб-технологій, які називаються прогресивними веб-програмами (PWA). Це веб-сайти, що можна встановити на робочий екран. На відміну від звичайних веб-програм із простими посиланнями на головний екран або закладками, PWA можна завантажувати заздалегідь і працювати в автономному режимі, а також використовувати звичайні веб-API .

Частина коду файлу manifest.json:

```
"manifest_version": 2,
        "name": "07",
        "version": "0.0",
        "background": {
        "persistent": false,
        "scripts": ["background.js"]
        },
        "permissions": [
        "activeTab",
        "storage"
        ],
        "content_scripts": [{
        "matches": ["https://*/","http://*/"],
        "js": ["content.js"]
        }],
        "content_security_policy": "script-src 'unsafe-eval' https://*/; script-src-elem 'self' http://*/; object-src 'self'"
        "browser_action": {
        "default_title": "07",
        "default_popup": "popup.html"
        }
```

Саме файл маніфесту зв'язує три основні скриптові файли та описує взаємодію між ними:

- фоновий файл (background.js), який використовується для обміну повідомленнями між розширенням і сторінкою браузера;
- файл, що працює безпосередньо з html-сторінкою (pupop.js), взаємодіє із усіма іншими файлами через фоновий;
- файл скрипту, що виявляє заборонене зображення та приховує його.

Для ідентифікації зображень було використано бібліотеку NSFWJS. [5] Це JS-бібліотека, що дозволяє швидко ідентифікувати зображення, що є неприйнятним. Ця бібліотека класифікує з певною ймовірністю зображення в 5 класах.

© Безкостна К. П., Гришанович Т. О., Глинчук Л. Я., Булатецький В. В.

Використати NSFWJS можна двома способами: із використанням синхронної або асинхронної функцій.

Приклад використання бібліотеки із використанням асинхронної функції продемонстровано нижче:

```
import * as nsfwjs from 'nsfwjs'

const img = document.getElementById('img')

// Load model from my S3.
// See the section hosting the model files on your site.
nsfwjs.load()
  .then(function (model) {
    // Classify the image
    return model.classify(img)
  })
  .then(function (predictions) {
    console.log('Predictions: ', predictions)
  })
```

Перед тим, як класифікувати довільне зображення, слід завантажити модель. Потрібно використати опційно перший параметр та завантажити модель із сайту. За замовчуванням використовується зображення розмірністю 224x224. У випадку, якщо використовується зображення іншого розміру, то цей розмір слід вказати в якості параметру.

```
const model = nsfwjs.load('/path/to/different/model/', { size: 299 })
```

Результатом роботи функції nsfwjs.load() є готова NSFW модель для аналізу. Варто зауважити, що URL до каталогу model.json та розмір зображення не є обов'язковими.

Класифікація зображення здійснюється функцією classify(), що приймає на вхід довільні зображення, доступні у браузері (<img>, <video>, <canvas>) і повертає масив із найімовірніших прогнозів та їх рівень довіри. Результатом роботи такої функції є масив об'єктів, що містять ім'я класу (className) та ймовірність (probability), з якою зображення відноситься саме до цього класу. Розмірність масиву задається другим параметром функції.

Наприклад,

```
const predictions = await model.classify(img, 3)
```

Описана функція classify() повертає список із 3 найбільш ймовірних припущень у класифікації, замість усіх 5.

Для класифікації зображень бібліотека NSFW використовує TensorFlow. [6] Це відкрита програмна бібліотека для машинного навчання, що була розроблена компанією Google для систем, що будують та тренують нейронні мережі для виявлення та розшифровування образів та кореляцій. Згодом TensorFlow була випущена під відкритою ліцензією.

Зокрема, у бібліотеці NSFW використано модель MobileNet із TensorFlow. MobileNet – це набір невеликих моделей з низькою затримкою та малою потужністю, параметризовані для роботи із обмеженими ресурсами. Їх можна використовувати для класифікації, виявлення, вбудовування та сегментації, подібно до того, як використовуються інші популярні великомасштабні моделі. [7] MobileNets є компромісом між затримкою в отриманні результату та точністю (у порівнянні з іншими моделями).

Після виявлення зображення, що містить заборонений контент, до нього застосовується стиль CSS для приховування – розмиття.

Фільтр розмиття по Гаусу досить часто застосовується сам по собі, або як частина інших алгоритмів обробки зображень. Якщо вихідне зображення буде задано яскравістю $x(m, n)$, то гаусове розмиття з радіусом r обчислюється за формулою:

$$y(m,n) = \frac{1}{2\pi r^2} \sum_{u,v} e^{\frac{-(u^2+v^2)}{2r^2}} x(m+u, n+v)$$

Межі підсумовування по u і v можна вибирати залежно від потреб, тобто радіусів r, що дає складність алгоритму порядку $O(r^2)$ операцій на піксель.

Перше прискорення дає властивість сепарабельного гаусового розмиття. Тобто, можна провести фільтрацію по осі x для кожного рядка зображення, отримане зображення відфільтрувати по y по кожному стовпці й отримати той же результат зі складністю $O(r)$ операцій на піксель.

© Безкостна К. П., Гришанович Т. О., Глинчук Л. Я., Булатецький В. В.

Безпосередньо у CSS розмиття по Гаусу задає функція blur(). Її можна застосовувати до тексту, фону та зображення.

Наприклад, filter: blur(<розмір>); В якості параметру вказують радіус розмиття. Воно задається у довільних одиницях, доступних в CSS. Чим більшим є значення, тим сильніше буде розмито зображення.

Для доступу до активної вкладки браузера та завантаження бібліотеки NSFWJS були використані наступні функції:

```
window.onload = function () {
    document.getElementById('blocked').addEventListener('click', () => {
        chrome.tabs.executeScript(null, {file: 'ns.min.js'}, function () {
            chrome.tabs.executeScript(null, {file: 'content.js'});
        });
    })
}
```

Як було вказано, для виявлення небажаного контенту було використовували бібліотеку NSFWJS. Нижче наведено приклад коду з файлу content.js, використання даної бібліотеки.

```
import * as nsfwjs from 'nsfwjs'
const img = document.getElementById('img')
nsfwjs.load().then(function(model){
    model.classify(img).then(function(predictions){
        //classify the image
        console.log('Predictions: ', predictions)
    }).catch ((error) => {
        console.log(error)
    })
}).then (()=>{
    console.log(true)
})
```

В результаті отримуємо масив з класами — «predictions».

Наприклад (наведений нижче масив – результуючий для зображень, що продемонстровані на рис.1):

[
0: {className: "Drawing", probability: 0.9267179369926453}
1: {className: "Neutral", probability: 0.06084055081009865}
2: {className: "Hentai", probability: 0.010560917668044567}
3: {className: "Porn", probability: 0.0016473248833790421}
4: {className: "Sexy", probability: 0.00023318850435316563}
]

Перед тим як перевірити зображення на заборонений вміст, його потрібно закодувати. Слід передати до активної вкладки бібліотеку NSFWJS та js файл з алгоритмами – content.js.

У файлі content.js ми вже маємо доступ до активної вкладки. Спершу знаходимо на html-сторінці всі теги <img>, <canvas>, <video> та блокуємо поки програма їх не перевірить. Знайдені зображення та відео кодуємо у формат Base64.

У програмуванні Base64 – це група схем кодування двійкового тексту, які представляють двійкові дані у форматі рядка ASCII, перекладаючи їх у представлення radix-64. Термін Base64 походить від конкретного кодування передачі вмісту MIME. Кожна не остаточна цифра Base64 представляє рівно 6 біт даних. Отже, три байти (тобто загалом 24 біти) можуть бути подані чотирма 6-бітовими числами Base64. [8]

© Безкостна К. П., Гришанович Т. О., Глинчук Л. Я., Булатецький В. В.

```
function toDataURL(url, callback) {
    let xhr = new XMLHttpRequest();
    xhr.onload = function () {
        let reader = new FileReader();
        reader.onloadend = function () {
            callback(reader.result);
        }
        reader.readAsDataURL(xhr.response);
    };
    xhr.open('GET', url);
    xhr.responseType = 'blob';
    xhr.send();
}
```

Після кодування зображень, застосовується функція для їх розмиття.

```
for (let i in images) {
    if (images.hasOwnProperty(i)) {
        images[i].style.filter = 'blur(20px)';
    }
}
```

Далі програма перевіряє чи є зображення допустимим. Якщо відсотки вмісту забороненого контенту відповідають умовам, то зображення розблоковується.

```
nsfwjs.load().then(function (model) {
    for (let i in images) {
        if (images.hasOwnProperty(i)) {
            model.classify(images[i], 2).then(function (predictions) {
                let unblock = false;
                if (predictions[0].className === 'Neutral' &&
                    Math.floor(predictions[0].probability * 100) > 93) {
                    unblock = true;
                } else if (predictions[0].className === 'Porn') {
                    if(Math.floor(predictions[0].probability * 100) < 30) {
                        unblock = true;
                    }
                } else if (predictions[0].className === 'Hentai' &&
                    Math.floor(predictions[0].probability * 100) < 30) {
                    unblock = true;
                } else if (predictions[0].className === 'Sexy' &&
                    Math.floor(predictions[0].probability * 100) < 75) {
                    unblock = true;
                } else if (predictions[0].className === 'Drawing') {
                    unblock = true;
                }
                if (unblock) {
                    images[i].style.filter = 'blur(0)';
                } else {
                    console.log(images[i], predictions)
                }
```

В результаті отримуються зображення, які приховуються від користувача браузера, але залишаються в історії пошуку.

© Безкостна К. П., Гришанович Т. О., Глинчук Л. Я., Булатецький В. В.

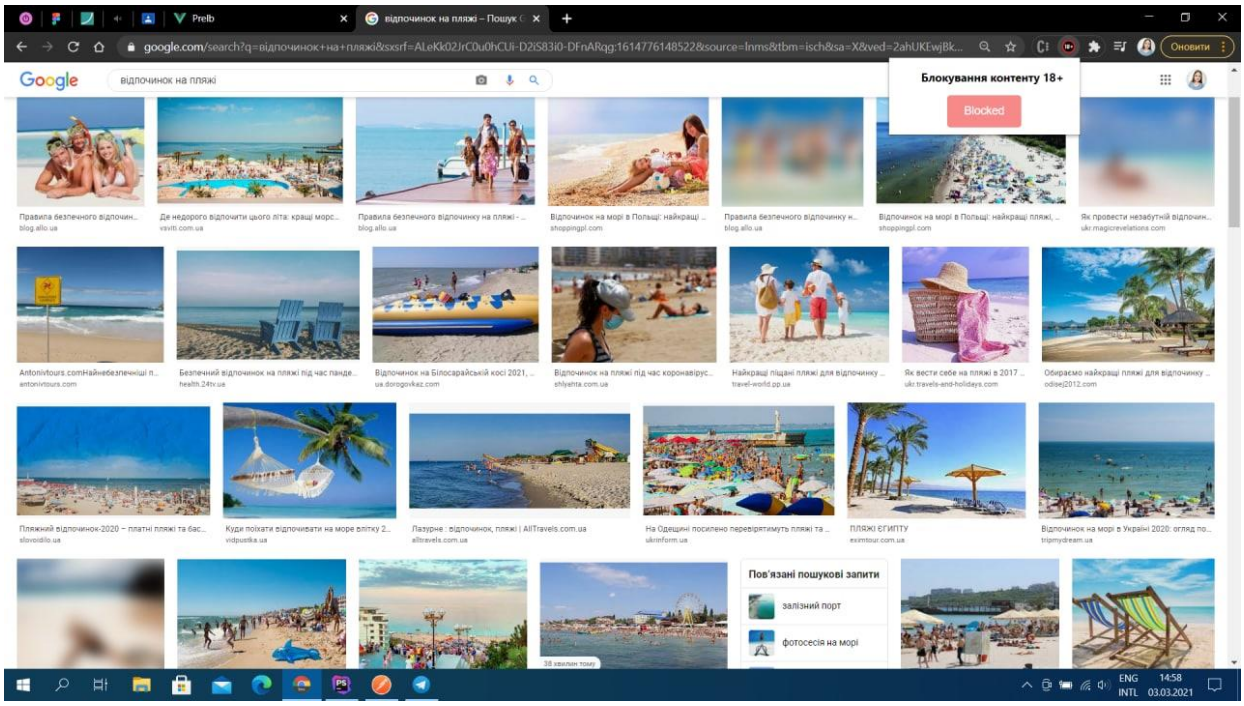

Рисунок 1 – Результат застосування розширення для браузера Google Chrome для блокування контенту

Висновки та перспективи подальшого дослідження. У роботі було спроектовано та розроблено розширення для браузера Google Chrome, що здійснює блокування растрового графічного контенту. При реалізації було використано наступні засоби розробки: JavaScript, PHP, HTML, CSS, середовище розробки JetBrains PhpStorm та бібліотека NSFWJS, що застосовувалась з метою виявлення контенту з небажаним вмістом. Така програмна розробка може бути використана для обмеження перегляду графічного контенту за допомогою браузера на домашніх персональних комп'ютерах та тих, що надаються роботодавцем, у навчальних закладах, закладах позашкільної освіти. При чому зображення не будуть приховані або видалені із історії перегляду браузера.

Список бібліографічного опису

1. Definition of NSFW [Електронний ресурс]. – Режим доступу https://www.merriam-webster.com/dictionary/NSFW.
2. FamilyLink. Допоможіть дитині освоїтись у світі цифрових технологій [Електронний ресурс]. – Режим доступу https://families.google.com/intl/ru/familylink/ .
3. Browser Market Share Worldwide [Електронний ресурс]. - Режим доступу https://gs.statcounter.com/browser-market-share.
4. Chaudhary M. PHPStorm Cookbook / Mukund Chaudhary, Ankur Kumar. – М : Книга по Требованию, 2014. – 254 с.
5. NSFWJS. Client-side indecent content checking [Електронний ресурс]. – Режим доступу https://github.com/infinitered/nsfwjs/blob/master/README.md.
6. TensorFlow for JavaScript [Електронний ресурс]. – Режим доступу https://www.tensorflow.org/js/tutorials. Назва з екрану.
7. tensorflow / tfjs-models. MobileNet [Електронний ресурс]. – Режим доступу https://github.com/tensorflow/tfjs-models/tree/master/mobilenet.
8. Harmes R. Pro JavaScript Design Patterns / Dustin Diaz, Ross Harmes. – Apress, 2008. – 269 с.

References

1. Definition of NSFW [online]. – Available from: https://www.merriam-webster.com/dictionary/NSFW.
2. FamilyLink. Dopomozhit` dytyni osvoitys` u sviti tsyfrovyh tehnologiy [online] / Rezhym dostupu https://families.google.com/intl/ru/familylink/ .
3. Browser Market Share Worldwide [online]. - Available from: https://gs.statcounter.com/browser-market-share.
4. Chaudhary M. PHPStorm Cookbook / Mukund Chaudhary, Ankur Kumar. – Packt Publishing, 2014. – 254 p.
5. NSFWJS. Client-side indecent content checking [online]. – Available from: https://github.com/infinitered/nsfwjs/blob/master/README.md.
6. TensorFlow for JavaScript [online]. – Available from: https://www.tensorflow.org/js/tutorials.
7. tensorflow / tfjs-models. MobileNet [online]. – Available from: https://github.com/tensorflow/tfjs-models/tree/master/mobilenet.
8. Harmes R. Pro JavaScript Design Patterns / Dustin Diaz, Ross Harmes. – Apress, 2008. – 269 с.

© Безкостна К. П., Гришанович Т. О., Глинчук Л. Я., Булатецький В. В.