ОСНОВНЫЕ ПРИНЦИПЫ СОЗДАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ СИСТЕМ РАСПОЗНАВАНИЯ ОБЪЕКТОВ

Джаматов Мустафа Хатамович
старший преподаватель кафедры цифровых технологий и информационной безопасности Академии МВД
Абдазимов Саидаминхожа Зоирхожаевич
преподаватель кафедры цифровых технологий и информационной безопасности Академии МВД

https://doi.org/10.5281/zenodo.14594352

Аннотация. *В данной статье рассматриваются основные принципы создания программного обеспечения для систем распознавания объектов, направленных на обеспечение кибербезопасности. Основой большинства современных систем являются алгоритмы машинного обучения (ML), в частности, методы глубокого обучения (DL), включая сверточные нейронные сети (CNN), специально разработанные для анализа изображений. Обсуждаются ключевые аспекты, такие как оптимизация производительности и точности распознавания объектов, которые зависят от архитектуры модели, качества данных и методов предобработки.*

Ключевые слова: *Системы распознавания объектов, машинное обучение, глубокое обучение, нейронные сети, аугментация данных, распознавание в реальном времени, безопасность и конфиденциальность.*

BASIC PRINCIPLES OF CREATING SOFTWARE FOR OBJECT RECOGNITION SYSTEMS

Abstract. *This article discusses the basic principles of creating software for object recognition systems aimed at ensuring cybersecurity. Most modern systems are based on machine learning (ML) algorithms, in particular deep learning (DL) methods, including convolutional neural networks (CNN) specifically designed for image analysis. Key aspects such as optimizing the performance and accuracy of object recognition, which depend on the model architecture, data quality, and preprocessing methods, are discussed.*

Keywords*. Object recognition systems, machine learning, deep learning, neural networks, data augmentation, real-time recognition, security and privacy.*

Системы распознавания объектов стали неотъемлемой частью современного программного обеспечения, применяясь в таких областях, как автономные транспортные средства, системы безопасности, медицинская диагностика и многое другое. Разработка программного обеспечения для таких систем требует учёта множества факторов и

специфических принципов, которые обеспечивают надёжность, точность и эффективность работы. В этой статье рассматривается ключевые принципы создания систем распознавания объектов.

Использование алгоритмов машинного обучения. Основой большинства современных систем распознавания объектов являются алгоритмы машинного обучения (ML), в частности, методы глубокого обучения (DL). Это включает использование нейронных сетей, таких как сверточные нейронные сети (CNN), которые специально разработаны для анализа изображений.

Принципы работы с машинным обучением:

- *Обучение на больших наборах данных*. Чтобы система распознавания объектов могла точно определять различные объекты, ей необходимо обучаться на большом количестве данных. Чем больше и разнообразнее данные, тем лучше будет обобщающая способность модели.
- *Аугментация данных*. Важной техникой является аугментация данных - искусственное увеличение набора данных путём изменения исходных изображений (например, поворотов, масштабирования, изменения цвета и других преобразований). Это помогает модели лучше адаптироваться к различным условиям.
- *Регуляризация и предотвращение переобучения*. Переобучение (overfitting) происходит, когда модель слишком хорошо запоминает тренировочные данные и плохо работает на новых данных. Для предотвращения этого используются такие методы, как регуляризация, добавление шума в данные, Dropout, и контроль сложности модели.

Оптимизация производительности и точности. Производительность и точность распознавания объектов зависят от многих факторов, таких как архитектура модели, качество данных и методы предобработки.

Принципы оптимизации:

- *Выбор правильной архитектуры нейронных сетей*. Различные архитектуры (например, ResNet, VGG, EfficientNet) имеют свои особенности и оптимизированы под различные задачи. Необходимо учитывать баланс между глубиной сети, её вычислительной сложностью и точностью.
- *Предобработка данных*. Обработка входных данных, таких как нормализация, выравнивание, устранение шумов, помогает улучшить качество работы модели и повысить её точность.
- *Постоянная проверка и настройка гиперпараметров*. Оптимизация гиперпараметров, таких как скорость обучения, размер батча и количество слоёв, играет ключевую роль в достижении высокой производительности модели.

• *Использование ускоряющих технологий*. Применение графических процессоров (GPU) и тензорных процессоров (TPU) значительно ускоряет обучение моделей и выполнение задач распознавания объектов в реальном времени.

Интеграция с реальными системами и работа в реальном времени. Для использования систем распознавания объектов в реальных приложениях (например, в автомобильных системах или камерах видеонаблюдения) важно обеспечить работу в режиме реального времени.

Принципы интеграции:

• *Оптимизация для реального времени*. Для приложений, работающих в режиме реального времени, необходимо минимизировать задержку и время обработки. Это можно достичь через использование облегчённых версий моделей, таких как MobileNet, или через оптимизацию модели путём применения квантования и сокращения весов.

• *Совместимость с аппаратными платформами*. Программное обеспечение должно быть адаптировано для работы на различных устройствах, от серверов до встраиваемых систем. Необходимо учитывать доступные вычислительные мощности и энергопотребление.

• *Надёжность и отказоустойчивость*. В реальных системах распознавания объектов важна надёжность работы в условиях помех, недостаточной освещённости, нестандартных позиций объектов и других неблагоприятных факторов. Для этого требуется применять различные техники фильтрации, устранения ложных срабатываний и постобработки результатов.

Тестирование и валидация. Тестирование программного обеспечения для распознавания объектов является критическим этапом разработки, так как оно позволяет оценить, насколько система эффективна и устойчива к различным условиям.

Принципы тестирования:

• *Разделение данных на тренировочные, валидационные и тестовые наборы*.

Тестирование модели на данных, которые она ранее не видела, является ключом к правильной оценке её эффективности.

• *Тестирование в реальных условиях*. Модели часто проходят хорошо на искусственно созданных тестовых данных, но могут столкнуться с трудностями в реальных условиях.

Поэтому важно проводить тестирование на данных из реальной среды.

• *Оценка на различных метриках*. Помимо общей точности (accuracy), важно учитывать такие метрики, как полнота (recall), точность (precision), F1-мера, время отклика и уровень ложных срабатываний.

Обеспечение безопасности и конфиденциальности. Системы распознавания объектов могут сталкиваться с задачами обеспечения конфиденциальности и безопасности, особенно в приложениях, где обрабатываются личные данные (например, в системах видеонаблюдения или биометрической идентификации).

Принципы безопасности:

• *Шифрование данных*. Необходимо обеспечивать шифрование как входных данных, так и результатов работы модели для защиты от несанкционированного доступа.

• *Анонимизация и приватность*. Важно внедрять технологии анонимизации данных, особенно в приложениях, связанных с идентификацией лиц, чтобы защитить персональные данные пользователей.

• *Защита от атак*. Модели машинного обучения могут быть уязвимы для атак, таких как adversarial attacks, где небольшие изменения в данных могут привести к неправильной классификации. Поэтому необходимо внедрять методы защиты, такие как детекция атак и укрепление моделей.

Математическая модель распознавания объектов основана на машинном обучении, особенно на свёрточных нейронных сетях (CNN)**.** В основе этого подхода лежат свёртки, активационные функции, функция потерь и метод оптимизации, используемые для обучения модели. Рассмотрим математическую модель распознавания объектов на примере алгоритма YOLO (You Only Look Once).

Входные данные и представление изображения. Пусть у нас есть изображение I размером $w \times h \times c$, где: w - ширина изображения, h - высота изображения, c — количество каналов (например, 3 для RGB).

Изображение подаётся в нейронную сеть, которая сначала разделяет его на сетку $S \times S$. В каждой ячейке сетки YOLO предсказывает несколько параметров.

*Выход сети и предсказания***.** Каждая ячейка сетки предсказывает:

• B ограничивающих рамок (bounding boxes), каждая из которых включает:

• координаты центра рамки (x,y);

• ширину и высоту рамки (w,h);

• вероятность объекта в рамке P_{obj};

• C классов объектов $p(c_1),p(c_2),\ldots,p(c_C)$.

Выход сети для каждой ячейки можно записать как вектор длиной $B \times 5 + C$, где 5 - это x,y,w,h и P_{obj}.

Для всего изображения размер выхода нейронной сети будет:

$$S \times S \times (B \times 5 + C)$$

Например, для сетки *13×13*, двух рамок на ячейку и *20* классов, выходной тензор будет иметь размер *13×13×(2×5+20)=13×13×30*.

Формулы для координат ограничивающих рамок. Каждая ячейка предсказывает координаты рамок относительно самой себя. Координаты центра рамки (x,y) нормализуются относительно ячейки сетки:

$$x=\sigma(t_x), \quad y=\sigma(t_y)$$

где t_x и t_y - выходные значения сети, и σ(·) - сигмоидная функция:

$$\sigma(z) = \frac{1}{1+e^{-z}}.$$

Ширина и высота рамки *w* и *h* нормализуются относительно всего изображения:

$$w = e^{t_w} \cdot p_w, \quad h = e^{t_h} \cdot p_h,$$

где t_w и t_h - выходные значения сети, а p_w и p_h - априорные размеры рамки (так называемые anchor boxes).

Функция потерь (Loss function). YOLO использует комплексную функцию потерь, которая учитывает несколько компонентов:

- Потери координат рамки (для *x,y,w,h*);
- Потери по вероятности объекта P_{obj};
- Потери по классам объектов.

Полная функция потерь включает в себя следующие составляющие:

Потери по координатам рамки. Для точных предсказаний координат рамки используется среднеквадратичная ошибка (MSE) между предсказанными и истинными значениями *(x,y,w,h):*

$$L_{\text{coord}} = \lambda_{\text{coord}} \sum_{i=1}^{S^2} \sum_{j=1}^{B} 1_{i,j}^{\text{obj}} \left((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2 \right),$$

где:

- $1_{i,j}^{\text{obj}}$ — индикатор, равный 1, если в ячейке *i* и рамке *j* есть объект, и 0 - если нет;
- λ_{coord} — коэффициент, регулирующий важность этой части функции потерь.

Потери по вероятности объекта. Потери по вероятности объекта также измеряются через среднеквадратичную ошибку:

$$L_{\text{obj}} = \sum_{i=1}^{S^2} \sum_{j=1}^{B} 1_{i,j}^{\text{obj}} (P_{\text{obj}} - \hat{P}_{\text{obj}})^2,$$

где P_{obj} - предсказанная вероятность объекта.

Потери по отсутствию объекта. Для ячеек, где нет объекта, потери тоже измеряются через MSE, но с другим весом:

$$L_{\text{noobj}} = \lambda_{\text{noobj}} \sum_{i=1}^{S^2} \sum_{j=1}^{B} 1_{i,j}^{\text{noobj}} (P_{\text{obj}} - \hat{P}_{\text{obj}})^2,$$

где λ_{noobj} — меньший коэффициент, чтобы не переоценивать влияние ячеек без объектов.

Потери по классам объектов. Для классов объектов используется кросс-энтропия**:**

$$L_{\text{class}} = \sum_{i=1}^{S^2} 1_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2,$$

где $p_i(c)$ - предсказанная вероятность класса c, а $\hat{p}_i$ - истинная вероятность класса.

Общая функция потерь:

$$L = L_{coord} + L_{obj} + L_{noobj} + L_{class}.$$

Обучение и оптимизация. Для оптимизации используется метод стохастического градиентного спуска (SGD) или его улучшенные версии, такие как Adam. Цель обучения — минимизировать функцию потерь L, чтобы улучшить предсказания модели на новых данных.

Эта математическая модель описывает, как алгоритм распознавания объектов, такой как YOLO, использует свёрточные нейронные сети для предсказания рамок и классов объектов на изображении. Основные составляющие — это предсказание координат, вероятностей и классов, с последующим вычислением функции потерь и оптимизацией её для улучшения работы модели.

REFERENCES

1. Goodfellow I., Bengio Y., Courville A. (2016). *Deep Learning*. MIT Press.
2. Krizhevsky A., Sutskever I., Hinton G. E. (2012). *ImageNet Classification with Deep Convolutional Neural Networks*. Advances in Neural Information Processing Systems (NIPS).

3. Redmon J., Divvala S., Girshick R., Farhadi A. (2016). *You Only Look Once: Unified, Real-Time Object Detection*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
4. He K., Zhang X., Ren S., Sun J. (2016). *Deep Residual Learning for Image Recognition*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
5. Deng J., Dong W., Socher R., Li L.-J., Li K., Fei-Fei L. (2009). *ImageNet: A Large-Scale Hierarchical Image Database*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
6. Zeiler M. D., Fergus R. (2014). *Visualizing and Understanding Convolutional Networks*. European Conference on Computer Vision (ECCV).
7. Ren S., He K., Girshick R., Sun J. (2015). *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI).
8. LeCun Y., Bengio Y., Hinton G. (2015). *Deep Learning*. Nature, 521(7553), 436–444.
9. Russakovsky O., Deng J., Su H., et al. (2015). *ImageNet Large Scale Visual Recognition Challenge*. International Journal of Computer Vision (IJCV), 115(3), 211-252.
10. Huang G., Liu Z., Van Der Maaten L., Weinberger K. Q. (2017). *Densely Connected Convolutional Networks (DenseNet)*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).