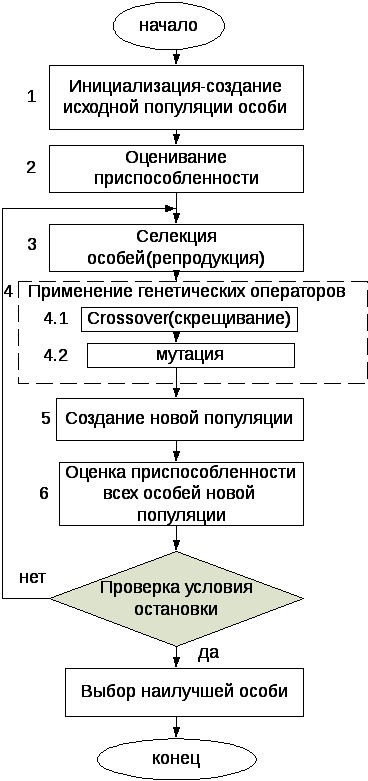
**«Эволюционные алгоритмы»**

Определение: представляет собой подмножество эволюционных вычислений, общий алгоритм метаэвристической оптимизации на основе популяции, который использует механизмы, вдохновлённые биологической эволюцией: размножение, мутация, рекомбинация и отбор (эволюционная стратегия), возможные решения играют в задаче оптимизации роль особей в популяции, а функция приспособленности определяет качество решений. Затем происходит эволюция популяции после многократного применения вышеуказанных операторов.

Эволюционные вычисления − в компьютерных науках представляют собой семейство алгоритмов глобальной оптимизации, вдохновлённых биологической эволюцией, являющиеся областью искусственного интеллекта и мягких вычислений (алгоритмы, которые терпимы к неточности, неопределённости, частичной правдивости и приближению: нейронные сети, нечеткая логика и эволюционные вычисления). В техническом плане они представляют собой семейство основанных на популяциях алгоритмов решения проблем методом проб и ошибок с метаэвристическим (стратегия, но неточная) или стохастическим (случайным) характером оптимизации.

Эволюционная стратегия −модель, которая использует естественные представления, зависящие от проблемы, и в первую очередь мутацию и отбор, в качестве операторов поиска, операторы применяются в цикле. Итерация цикла называется поколением. Последовательность поколений продолжается до тех пор, пока не будет выполнен критерий завершения. Для пространств поиска с действительными значениями мутация выполняется путём добавления нормально распределенного случайного вектора.

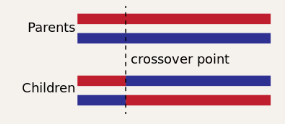
Имеет следующую последовательность:



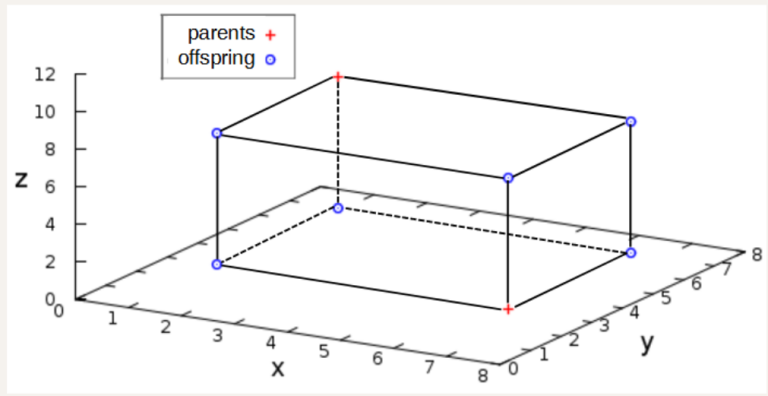
Компоненты:

1. Начальная популяция (генерируется автоматически или подбирается на основе определённых признаков).
2. Операторы скрещивания:
   1. Операторы на основе разрезов:

* одноточечный разрез, который ставится в случайном месте одинаково для обеих хромосом, после чего на его основе делается перестановка элементов;



* многоточечный разрез, аналогичен одноточечному с поправкой на то, разрезов больше одного.
  1. Операторы на основе вероятности:
* равномерный кроссовер, в котором у дочерней хромосомы поочерёдно определяется вероятность выбора гена от каждого из родителей, что даёт равный шанс перехода генов каждого из родителей;
* дискретная и промежуточная рекомбинации, которые позволяют выбрать гены дочерней хромосомы путём разложения хромосомы размером n в n-мерное пространство для случайного выбора вершин, в первом случае выбираются только состояния в вершинах многогранника, во втором − ещё и промежуточные состояния (актуально для действительных чисел).



1. Операторы мутации:
   1. операторы обмена (перестановка элементов внутри хромосомы):

* простой обмен (перестановка двух случайных генов);
* скрэмблер (перестановка элементов в определённой области случайным образом);
* инверсная (зеркальный переворот всех элементов в хромосоме);
  1. операторы сброса значения:
* битовые переворот (переворот 0 на 1 и 1 на 0);
* случайный сброс (выбирается случайное число из допустимого диапазона).
  1. мутация на основе законов распределения:
* мутация на основе закона распределения (без учёта ограничений) случайного величины относительно диапазона значений или их величины;
* мутация с учётом ограничений, в которой в наименьшие изменения наиболее вероятны.

1. Операторы отбора:
   1. выбор колеса рулетки (на основе фитнеса и равномерный);
   2. выбор на основе турнира;
   3. элитарный отбор;
   4. отбор на основе градиента (отбор Больцмана).
2. Фитнес-функция (функция пригодности, особый тип целевой функции, который используется для обобщения в виде единого показателя качества того, насколько близко данное проектное решение к достижению поставленных целей). Целевая функция − вещественная или целочисленная функция нескольких переменных, подлежащая оптимизации в целях решения некоторой оптимизационной задачи, может быть как в виде функции потерь (которую надо минимизировать), так ей обратных (которые надо максимизировать): функция вознаграждения, функция прибыли, функция полезности, функция пригодности.

Особенности:

Ограниченная структура решений (в задачах программирования означает, что программа имеет либо ограничения по памяти, либо по времени, либо по структуре кода), в более простом варианте обозначает то, что ограничены размеры популяции.

Основным оператором является оператор мутации для повышения разнообразия представителей популяции (может применятся несколько), как правило, работает с действительными значениями в широком диапазоне.

Основные задачи: аппроксимация, симуляция и оптимизация.

**«Генетические алгоритмы»**

Определение: конкретизированный вид эволюционных алгоритмов, который обычно используется для создания высококачественных решений задач оптимизации и поиска, полагаясь на биологически вдохновлённые операторы.

Особенности в отличиях:

Неограниченная структура решений.

Основным оператором является оператор скрещивания для регуляризации пространства поиска решений.

Основные задачи: поиск решений и оптимизация.

**«Многослойные нейронные сети»**

Компоненты:

1. Архитектура нейрона:
   1. количество входов;
   2. количество весов;
   3. функция активации;
   4. количество выходов.
2. Архитектура нейронной сети:
   1. количество слоёв нейронной сети;
   2. количество связей между нейронами в сети (полносвязная и неполносвязная);
   3. количество нейронов в слое и их функции активации.
3. Оптимизатор нейронной сети (алгоритмы, которые используются для оптимизации параметров нейронной сети, таких как веса и смещения, с целью уменьшения степени ошибки модели и улучшения её качества):
   1. **метод градиентного спуска**, который заключается в изменении параметров нейронной сети в направлении, противоположном градиенту функции потерь, градиент функции потерь определяет направление, в котором следует изменять параметры модели, чтобы уменьшить ошибку.
   2. **метод обратного распространения ошибки**, который заключается в следующих шагах:
4. прямое распространение сигнала через нейронную сеть, начиная с входного слоя и заканчивая выходным слоем, каждый нейрон вычисляет свой выход на основе входных данных и активационной функции.
5. вычисляется ошибка на выходном слое с помощью функции потерь, которая сравнивает выходные значения сети и желаемые значения, определённые в обучающих данных.
6. ошибка распространяется обратно через сеть, начиная с выходного слоя и двигаясь к входному слою, каждый нейрон получает градиент ошибки от вышестоящего нейрона и обновляет свои веса с помощью градиентного спуска.
7. шаги 1-3 повторяются для каждого обучающего примера в наборе данных, этот процесс называется эпохой обучения.
8. после завершения всех эпох обучения, модель может быть использована для предсказания выходных значений на новых данных.
9. Функция потерь и метрики качества.
   1. Функции потерь позволяют понять, насколько ожидаемые результаты отличаются от полученных:

* **Простая ошибка** как разница между ожиданием и результатом:

loss = y - f(x),

где y - ожидаемый результат

x - полученный результат;

* **Функция ошибки классификации**, которая возвращает 0, если предсказание модели верно и 1, если неверно:

CLF = 1 - (accuracy / 100)

где:

accuracy - процент верных предсказаний модели;

* **Среднеквадратичная ошибка** (Mean Squared Error, MSE) − это наиболее распространенная функция потерь для задач регрессии, она определяется как средняя арифметическая квадратичных разностей между предсказанными и истинными значениями,

MSE = 1/n \* sum((y - ŷ)^2)

где:

n - количество наблюдений или размер выборки

y - истинное значение целевой переменной

ŷ - предсказанное моделью значение целевой переменной

MSE вычисляет среднее арифметическое квадратов отклонений прогнозов модели от истинных значений. В результате, чем меньше значение MSE, тем лучше модель предсказывает значения целевой переменной;

* **Кросс-энтропия** (Cross-Entropy Loss) − это функция потерь, которая используется для задач классификации, она определяется как отрицательная взвешенная сумма логарифмов предсказанных вероятностей классов,

формула Кросс-энтропии (Cross-Entropy Loss) для бинарной классификации выглядит следующим образом:

CE = - (1/N) \* sum(y \* log(ŷ) + (1 - y) \* log(1 - ŷ))

где:

N - количество наблюдений или размер выборки

y - истинное значение класса (0 или 1)

ŷ - предсказанная моделью вероятность класса 1

* 1. Метрика качества позволяет определить, насколько хорошо модель обучена на заданной выборке данных:
* **Accuracy** (точность) − это доля правильных ответов, которые модель даёт на тестовой выборке данных.
* **Precision** (точность) −это доля правильных предсказаний положительного класса (true positives/(true positives + false positives)).
* **Recall** (полнота) −это доля правильно предсказанных положительных значений относительно всех реальных положительных значений (true positives/(true positives + false negatives)).

1. Регулязатор−техника, которая помогает предотвратить переобучение модели на тренировочных данных, она включает в себя добавление дополнительных условий на параметры модели во время обучения, чтобы предотвратить экстремальные значения весов и смещений, которые могут привести к переобучению.
   1. **L1** регуляризация −добавляет штраф относительно суммы абсолютных значений весов нейронной сети (L1 loss = λ \* Σ |w|, где w - весовые коэффициенты модели, λ - коэффициент регуляризации.).
   2. **L2** регуляризация −добавляет штраф относительно суммы квадратов весов нейронной сети (L2 loss = λ \* Σ w^2, где w - весовые коэффициенты модели, λ - коэффициент регуляризации).
   3. **Dropout** −случайным образом отключает некоторые нейроны в каждом слое во время обучения сети, чтобы предотвратить переобучение.

Функции активации позволяют нейронам в сети передавать информацию и выполнять необходимые преобразования входных данных:

**Ступенчатая функция активации** (step function) − это простая нелинейная функция, которая имеет значение 1, если её аргумент больше или равен 0, и 0 в противном случае. Формально, ступенчатую функцию активации можно определить следующим образом:

f(x) = 1, x >= 0

f(x) = 0, x < 0

Также существует вариант ступенчатой функции активации, которая имеет значение 1 при x строго больше 0:

f(x) = 1, x > 0

f(x) = 0, x <= 0

Ступенчатая функция активации может использоваться в некоторых типах нейронных сетей, например, в однослойных персептронах, ступенчатая функция активации нечасто применяется из-за того, что она не учитывает относительную силу входных сигналов, которые приходят на нейрон, также она дискретна и недифференцируема, её более популярными аналогами являются сигмоидальная функция или ReLU.

**Сигмоидная функция** (sigmoid) − функция, которая возвращает значения между 0 и 1. Она имеет S-образную форму, что позволяет ей выполнять бинарную классификацию входных данных, её главный недостаток - тенденция к насыщению (то есть, когда функция принимает значения близкие к 0 или 1, её градиент сильно затухает).

**Гиперболический тангенс** (tanh) − эта функция также имеет S-образную форму, но возвращает значения в диапазоне от -1 до 1. В отличие от сигмоидной функции, гиперболический тангенс обладает нулевым центром (среднее значение равно 0), что делает его полезным для центрирования данных.

**ReLU** −(Rectified Linear Unit) − Эта функция возвращает значение равное 0, если вход меньше или равен 0, и значение входа, если он больше нуля. Функция ReLU очень популярна в нейронных сетях, потому что она проста, быстра и обычно работает лучше других функций активации на практике. Она также помогает бороться с проблемой затухания градиентов.

**Leaky ReLU** (LReLU)−эта функция похожа на выполнение ReLU за исключением того, что она имеет небольшой наклон для отрицательных значений. Это помогает избежать изменения знака градиента, что может улучшить скорость сходимости.

**Softmax** − Эта функция используется обычно в последнем слое нейронной сети, чтобы перевести выходные значения в вероятностное распределение. Она принимает на вход вектор значений и возвращает вероятности находящиеся в интервале [0, 1], сумма которых равна 1. Softmax позволяет выбирать из нескольких классов, исходя из наиболее вероятного класса.

**Maxout** − Эта функция возвращает максимальное значение из нескольких линейных функций. Она может быть использована для отображения более сложных функций, чем простые линейные модели.

Особенности:

**Многослойность**:

В нейронной сети много слоёв используются для решения более сложных задач, которые требуют высокой степени абстракции и обобщения.

Каждый слой в сети отвечает за определённый аспект обработки данных. Некоторые слои могут обрабатывать простые характеристики, такие как грани и контуры, в то время как другие могут анализировать более сложные взаимосвязи между наборами характеристик.

Применение многих слоёв позволяет моделировать более сложные функции, которые не могут быть представлены более простыми моделями. Это может помочь в обработке различных типов данных, таких как текст, изображения, аудио и видео.

**Количество нейронов в слое**:

Увеличение количества нейронов в слоях нейронной сети позволяет улучшить её способность к обработке информации и улучшить точность предсказания.

При добавлении нейронов в слои, нейронная сеть имеет больше данных, которые могут быть использованы для обучения, что позволяет лучше извлекать закономерности и паттерны в данных.

Больше нейронов также позволяют моделировать более сложные и абстрактные зависимости между переменными, что помогает в обработке более сложных типов данных.

Однако, увеличение количества нейронов может также привести к переобучению.

**Выбор функции активации:**

Однако, выбор функции активации зависит от конкретной задачи и оптимального результата, а также от архитектуры нейронной сети. Некоторые функции могут быть более подходящими для определённых типов данных или задач, например, функция sigmoid лучше подходит для бинарной классификации.

Кроме того, в некоторых случаях могут быть использованы нестандартные функции активации, разработанные специально для решения определённых задач. Например, любые непрерывные, монотонно возрастающие и дифференцируемые функции могут быть использованы в качестве функции активации. Таким образом, выбор оптимальной функции активации играет важную роль в достижении результатов.

**«Многоагентные системы»**

СМОТРЕТЬ ФАЙЛ С ЗАДАНИЕМ К ПРАКТИЧЕСКОЙ РАБОТЕ №1!