과제 명세

1. 내용

- 개인이 임의의 클래스를 정의 (ex. 도형 클래스, 좌표 클래스, 성적 클래스 등)
- 정의한 클래스를 설계하고 구현
- 구현한 클래스에 대한 테스트

2. 구현 조건

- 멤버변수와 멤버함수의 정의
- 생성자, get/set 멤버함수, const 멤버함수, 인라인 함수 구현
- 함수 오버로딩 또는 기본 매개변수 사용
- 테스트의 경우 객체 3개이상 사용하여 모든 멤버함수 이용한 결과 제시

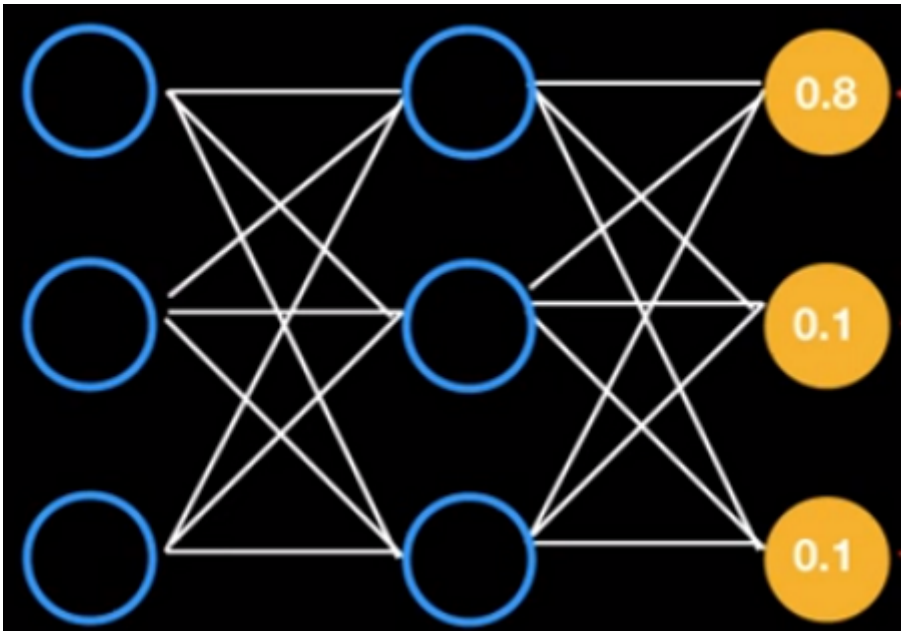
3. 제출

- 클래스 헤더파일 및 소스파일
- 테스트 파일
- header file(.h) 1개, source file(.cpp) 2개, 총 3개 파일 압축(학번_이름.zip)하여 업로드

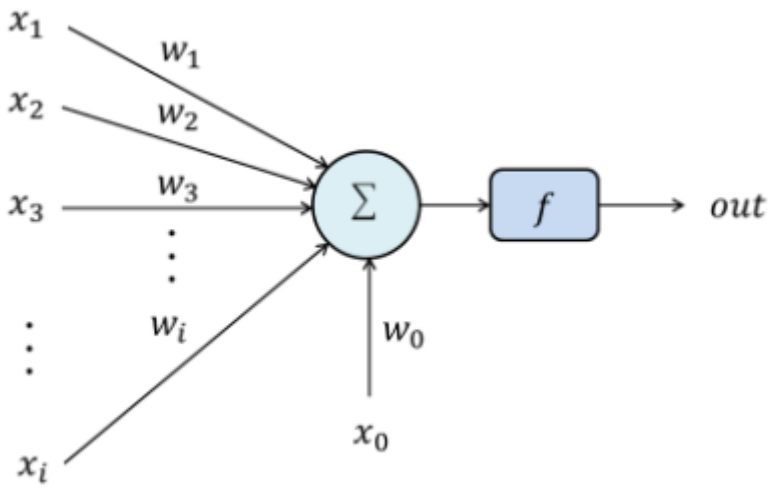
4. 기한

- 2020.04.13.월까지

과제 풀이



내가 구현한 것은, 인공 신경망의 기초가 되는 "퍼셉트론" 이다.



- f : activation function 은 sigmoid 만 구현했다.
- x0 : bias 는 구현하지 않았다.
- w : 모든 가중치 w가 0.1 로 초기화되는 constant 만 구현했다.
- 여러 개의 퍼셉트론을 연결하는 것은 구현하지 않았다.

실행 방법

노드를 생성한다.

- Perceptron node(3, "sigmoid");

input data 를 정의한다. input data 는 double 형이어야 한다.

- double input[3] = { 1.5, 2.3, 6.9 };

노드에 데이터를 넣어준다.

- node.set_input(input);

가중치를 초기화한다.

- node.WeightInitializer("constant");

연산을 시작한다.

- node.Run();

연산 결과 및 정보들을 받아본다.

- node.ShowAllForDebug();

유의사항

이름 규칙

참조

- NHN/C++ 코딩 규칙
- 안용학 교수님이 수업에서 언급했던 규칙
- Google C++ Style Guide

파일명

- 지금 하고 있는 프로젝트의 컨벤션, 안용학교수님의 컨벤션에 따른다.
- 클래스를 설계하는 경우, 파일 앞에 대문자 C 를 붙인이고, 대쉬 (-) 를 붙이고, 대표 class 이름으로 사용하며, C 를 제외하고는 소문자와 언더바를 사용한다. ex : **C-rect_base**
- 테스트용 (실행 파일) 의 경우, Test- 를 가장 처음에 포함한다. ex : **Test-rect_base**
- 파일 이름에 대쉬(-) 를 두 개 이상 사용하지 않는다.

함수명

- 일반적인 함수는 대문자로 시작하며, 각 새로운 단어마다 대문자를 사용한다. 언더라인은 사용하지 않는다. ex : **MyExcitingFunction()**
- 접근자와 수정자(get, set)는 변수 이름과 일치시킨다. ex : **set_my_exciting_member_variable()**

- True/False 값을 return 하는 경우, 함수 이름은 is 혹은 has 로 시작한다. ex : **IsHungry()**
- private 함수 이름은 언더바(_) 로 시작한다. ex : **_DontTouchMe()**

타입명

- 타입명은 대문자로 시작하며, 각 새로운 단어마다 대문자를 갖으며 언더라인을 사용하지 않는다. ex : **MyRectangle**

변수 및 상수명

- 변수명은 모두 소문자이며 단어 사이에 언더라인을 사용한다.
- 클래스 멤버 변수는 언더라인으로 끝난다. ex : **my_exciting_local_variable_**
- 이름은 가능한 설명적으로 짓는다. 공간 절약이 중요한 게 아니라, 코드를 즉시 보고 이해할 수 있어야 한다. ex : **num_completed_connections**
- 모호한 약어나 의미를 알 수 없는 임의의 문자를 사용하지 않는다. ex : **nerr** (?)
- 구조체의 데이터 멤버는 일반적인 변수처럼 이름을 짓는다. 클래스처럼 언더라인으로 끝나지 않는다.
- 전역 변수는 특별한 요구사항이 없으며, 거의 사용을 하지 않는다. 만약 사용한다면, g_로 시작하거나 로컬 변수와 구별되는 표시를 한다.
- 상수는 k로 시작하며 대소문자를 섞어서 사용한다 : ex : **kDaysInAWeek**

기타

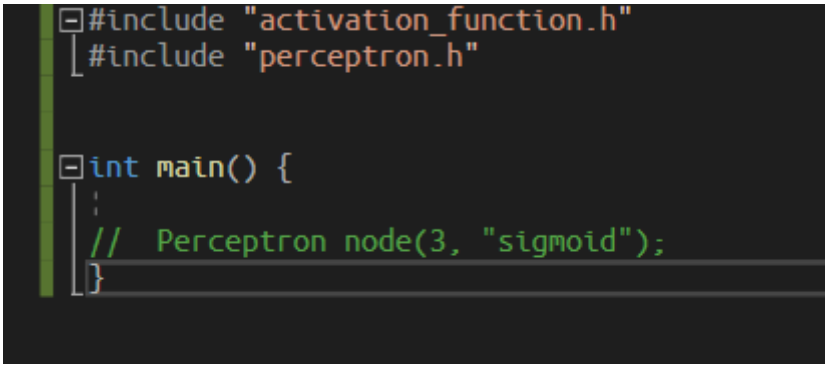
- 들여쓰기는 Tab 을 사용한다.
- 간단한 생성자 초기화는 콜론 초기화로 한다.
- 이항 연산자 (=, >, <, 등..) 앞과 뒤에 공백을 제공한다. ex : **a = b + c**
- 단항 연산자 앞과 뒤에 공백을 제공하나, (A++), [--BB], {--KK}와 같이 사용할 때는 공백이 없어도 좋다.
- 일부 연산자(“ , “, “ ; “)는 연산자 뒤에 공백을 제공해야 한다. ex : **for(i = 0; i < 3; i++)**
- brace({)는 분리된 라인에 작성한다.

```
class People
{
    // 내용
}

void main()
{
    // 내용
}

struct DataStructure
{

}
```

참고한 내용

Static

내 클래스에서 전역공간에 있는 변수를 찾으려고 static 키워드를 사용해 봤는데 잘 되지 않았다. 그런데 뒤지던 끝에, 내 클래스 안에 static 멤버 변수를 넣으면, 전역공간에 존재하는 변수를 찾는다는 것을 알게 되었는데, 클래스에서 선언만 하는 것만으로는 생성자가 호출이 안되는 모양이다.



이런 코드에서도, static 이 들어 있는 클래스를 먼저 뒤져서 먼저 컴파일을 하고 전역공간에 올려 두는 모양이다. main 함수에는 아무것도 없는데 컴퓨터가 무엇인가 일을 한다.

왜 이런 문제가 나오는지는 과제를 하느라 아직 이해하지 못했지만.. 디버깅을 해 보면, 나는 호출하지도 않은 클래스를 먼저 돌면서 정의를 하는 것 같다.

곧 배우리라 믿는다.

아래처럼 static으로 선언된 변수를 찾지 못할 경우

```
=== CTest.h =====
#ifndef __TEST_H__
#define __TEST_H__
class CTest
{
public:
 CTest(){}
 ~CTest(){}
 static bool Test();
private :
 static bool m_bA ;
};
#endif

=== CTest.cpp =====
void CTest::Test()
{
 m_bA = true ;
}
```

이렇게 바꾸어주면 된다.

```
=== CTest.cpp =====
void CTest::Test()
{
 m_bA = true ;
}
bool CTest:::m_bA = false;
```

이런식으로 static 변수는 class 내부에 선언을 하게되면
이렇게 Class 밖에서 정의가 되어야 한다.
이때 필요하면 초기화도 함께 한다.

function pointer

이 함수 포인터 사용하면서 애를 많이 먹었는데.

그냥 함수 포인터 주소만 덩그러니 있으면 못 찾나보다.

아무래도 객체마다 가상메모리를 주는 방식인가, 클래스 메소드 포인터 주소를 넘겼는데 그것을 계속 인식을 못 하는 것 같다. 왜 그런지는 아직 모르겠지만, 아래 애들 다 시도해보면서 해결했다.

https://stackoverrun.com/ko/q/11059111

https://stackoverflow.com/questions/990625/c-function-pointer-class-member-to-non-static-member-function