OPERATING INSTRUCTIONS

The operation principle

- Reference convolution parameters, including results, are calculated and stored in the program code.
- After verifying the configuration, which specifies whether to run the correctness checking module, the components for calculating the specified reference convolutions are initialized.
- Convolutions are computed using the basic algorithm.
- The obtained results for each of the convolutions specified by the correctness checking module are compared to the reference results.
- Convolutions are computed using optimized algorithms. In this case, correctness checking is performed not against reference data but against the result of the convolution using the basic algorithm, as its correctness has already been confirmed.
- The time elapsed since the start is checked. If less than a second has passed, the calculation process is repeated.
- After one second or more, the number of computed convolutions is recorded.
- The comparison results and the calculated number of convolutions per second are displayed on the console or in a log file, depending on the configuration settings.

To run the program, you must execute the command
./convbench -f=1

convbench has the following flags:

-h, --help, -? - call this help with subsequent exit.
-q, -quiet Quiet mode. Only the result of the convolution calculation is displayed.
-v, -verify - testing the correctness of the folds.
-l="log.txt", -log="log .txt" - log the output to the specified file - log.txt. The file name can be in quotes (if it contains spaces) or without them. Console output is duplicated in this file.
-j="config.json", -config="config.json" - load configuration from the specified file - config.json. The file name can be in quotes (if it contains spaces) or without them. This file specifies the parameters of the utility.
-t=ZZZ, -type=ZZZ - selection of calculated data types. Several types may be
used. Valid data types:
• f64 – double (signed floating point 64 bit),
• f32 – float (signed floating point 32 bit),
• i32 – int (signed integer 32 bit),
• i8 – char (signed integer 8 bit).
If this parameter is present, then it applies to all folds with no specified data types. If the parameter is not specified anywhere, then the default value f32 is selected for folds without specifying a type.
For example: "-t=f32i32i8".
This parameter can only be present in one instance.
-z=X, -optimize=X – optimization (X). May be:
• 0 – no optimization, standard algorithm;
• 1 – SSE optimization (128 bits), advanced algorithm;
• 2 - AVX optimization (256 bits), advanced algorithm;
• 5 - AVX512 optimization (512 bits), advanced algorithm.If this parameter is present, then it applies to all folds with no optimization specified.
Several optimization options can be specified, for example "-z=0125".
Predefined convolutions
-f=ZZZ, -factory=ZZZ - select predefined folds (one or more values).

Convolutions are specified by numbers:
• 1 – LeNet-5 input 32*32 core 5*5, offset 1
• 2 – AlexNet input 224*224 core 11*11, shift 4, with addition
• 3 – VGG-19 input 224*224 core 3*3, shift 2, with addition
• 4 - ResNet input 224*224 core 7*7, shift 2, with addition
• 5 – SRCNN input 32*32 core 9*9, shift 1, with complement
You can specify several options for convolutions.
For example: "-f=145" means: selection of 1,4,5 fold from the list with the
default type (if it is not specified by a separate parameter).
Flags for manually specifying convolutions
The following flags are used to set convolutions manually:
-rH*W, -tensorH*W – dimension of the input tensor. Where H is the height, W is
the width. H and W can be set between 16 and 512.
If a single value is specified, such as -r448, it means that both H and W are set to 448.
This parameter is required.
This parameter can only be present in one instance.
-kH*W -kernelH*W – kernel dimension. Where H is the height, W is the width. H
and W can be set between 3 and 31. The default is H and W = 3.
If a single value is specified, such as -k5, it means that both H and W are set to 5.
This parameter can only be present in one instance.
-sH*V, -strideH*V - shift (stride) horizontally and vertically. If one number is specified, then it applies to two axes. N - from 1 (default) to 512. The shift is reduced automatically by the size of the tensor).
If a single value is specified, such as -s10, it means that both H and V are set to 10.
This parameter can only be present in one instance.
-pH*V, -paddingH*V – padding with zeros. Can be 0 (no padding) or 1 (padding).
If one digit is specified, then it is applied to two axes. There is no shift by default.
If a single value is specified, such as -p1, it means that both H and V are set to 1.
This parameter can only be present in one instance.
-iZ, -inputsZ – number of input channels (Z). Can be from 1 to 256.
This parameter can only be present in one instance.
-oZ, -outputsZ – number of outputs (Z). Can be from 1 to 256.
This parameter can only be present in one instance.
-cR,K,S,P,I,O,T,Z -convR,K,S,P,I,O,T,Z - selection of used convolutions (one or more values). Where:
• R - H*W – input tensor dimension;
• K - H*W – kernel dimension;
• S - H*V – shift (stride);
• P - H*V – zero padding (padding);
• I – number of input channels;
• O – number of exits;
• T – used data types;
• Z - optimization.
In this case, the required parameters are the size of the tensor and the kernel, the rest can be omitted (the default values will be used).
This parameter can be present in several instances, i.e. you can specify several different rollups. For example: "-c224*224.3*3.1*1.1 -c256,11.2.0.1.64,f32i8.0125"

Thus, launching the program with predefined convolutions might look like this: ./convbench -f=12345 -t=i8 -z=0125

Thus, launching the program with manually set convolutions might look like this:

./convbench -r256 -k11 -s2 -p0 -i1 -o64 -t=f32i8 -z=0125
or
./convbench -c256,11,2,0,1,64,f32i8,0125

Example of a configuration JSON file:

```
{
   "config":{
      "quiet" : false,
      "log" : "newlog.txt",
      "verify" : 0.05
   },
   "conv":[
      {
         "tensor": "5*5",
         "kernel": "3*3",
         "stride": "1",
         "padding": "1",
         "inputs": "1",
         "types": "f32",
         "note": "test conv"
      },
      {
         "tensor": "224*224",
         "kernel": "3*3",
         "stride": "2*2",
         "inputs": "3",
         "types": "i32"
      },
      {
         "tensor": "224*224",
         "kernel": "3*3",
         "stride": "1*1",
         "outputs": "16"
      }
   ]
}
```

In the "**config**" section::

- "quiet" - quiet mode, equivalent to the "-q" flag. It can have the values false or true.
- "log" - indicates whether logging to a file is enabled, equivalent to the "-l" flag. The value corresponds to the name of the log file.
- "verify" - indicates whether to test the correctness of the convolution, equivalent to the "-v" flag.

In the "**conv**" section, an array of convolutions is defined. For each convolution, the following fields are used: "tensor," "kernel," "stride," "padding," "inputs," "outputs," and "types," which

correspond to the analogous command-line keys. If these fields are absent, their default values are used.
The "note" field provides a textual description of the specified convolution (displayed during calculation).

To check the use of the library, an example of the implementation of mathematical operations using available processor operations (SSE, AVX, AVX512) is given.

For this:

1) Go to example directory
cd ../example

2) Build the example
mkdir build
cd build
cmake ..
make

3) The executable file vecmulsum will appear in the build directory

4) To run the example, you need to run the command
./vecmulsum -t=i8 -i=../random_int8_1.bin -i=../random_int8_2.bin

vecmulsum has the following flags:

-h, --help, -? - call this help with subsequent exit.
-q, -quiet Quiet mode. Only the result of the calculation is displayed.
-t=Z, -type=Z - selection of data type to be calculated. Valid data types:
• f64 – double (signed floating point 64 bit),
• f32 – float (signed floating point 32 bit),
• i32 – int (signed integer 32 bit),
• i8 – char (signed integer 8 bit).
-i=filename, -infile=filename - select input files (filename).
Two names (in two parameters) of binary files must be specified,
each of which stores vector data of the desired type.

In binary example files, sequences are saved for various data types and they should only be used with the corresponding keys that describe that data type. For instance, the "random_float*" files are used only with the -t=f32 key, while the "random_int8*" files are used only with the -t=i8 key.

ИНСТРУКЦИЯ ИСПОЛЬЗОВАНИЯ

Принцип действия

- Эталонные параметры свёрток (включая результат) рассчитаны и сохранены в программном коде.
- После проверки конфигурации, где указана необходимость запуска модуля проверки корректности результатов свёртки, инициализируются компоненты расчёта свёрток заданных эталонных свёрток.
- Производится расчёт свёрток базовым алгоритмом.
- Полученные результаты для каждой из свёрток заданных модулем проверки корректности сравниваются с эталоном.
- Производится расчёт свёрток алгоритмами с оптимизацией. В этом случае проверка результата корректности каждой свёртки проводится не с эталонными данными, а с результатом выполнения свёртки с использованием базового алгоритма, поскольку правильность его работы ранее уже подтверждена.
- Проверяется прошедшее с момента фиксации время. Если прошло менее секунды процесс расчёта повторяется.
- Через секунду или более фиксируется число рассчитанных свёрток.
- Результат сравнения и рассчитанное число свёрток за одну секунду выводится на консоль или в лог-файл, в зависимости от настроек конфигурации.

Для запуска программы необходимо выполнить команду
./convbench -f=1

convbench имеет следующие флаги:

-h, --help, -? - вызов данной справки с последующим выходом.
-q, -quiet – тихий режим. Выводится только результат расчёта свёртки.
-v, -verify – тестирование корректности выполнения свёрток.
-l=“log.txt“, -log=“log .txt“ - протоколирование вывода в указанный файл – log.txt. Имя файла может быть в кавычках (если содержит пробелы) или без оных. В этот файл дублируется вывод консоли.
-j=“config.json“, -config=“config.json“ - загрузка конфигурации из указанного файла – config.json. Имя файла может быть в кавычках (если содержит пробелы) или без оных. В этом файле задаются параметры работы утилиты.
-t=ZZZ, -type=ZZZ - выбор обсчитываемых типов данных. Может быть использовано несколько типов. Допустимые типы данных:
• f64 – double (знаковое с плавающей точкой 64 bit),
• f32 – float (знаковое с плавающей точкой 32 bit),
• i32 – int (знаковое целое 32 bit),
• i8 – char (знаковое целое 8 bit).
Если данный параметр присутствует, то он применяется ко всем свёрткам с не заданными типами данных. Если параметр нигде не указан, то для свёрток без указания типа выбирается значение по умолчанию f32.
Например: «-t=f32i32i8».
Данный параметр может присутствовать только в одном экземпляре.
-z=X, -optimize=X – оптимизация (X). Может быть:
• 0 – без оптимизации, стандарный алгоритм;
• 1 – SSE оптимизация (128 бит), усовершенствованный алгоритм;
• 2 – AVX оптимизация (256 бит), усовершенствованный алгоритм;

• 5 – AVX512 оптимизация (512 бит), усовершенствованный алгоритм.Если данный параметр присутствует, то он применяется ко всем свёрткам с
не заданной оптимизацией.
Может задаваться несколько вариантов оптимизации, например «-z=0125».
Преднастроенные свёртки
-f=ZZZ, -factory=ZZZ - выбор предварительно заданных свёрток (одно или несколько значений). Свёртки задаются по номерам:
• 1 – LeNet-5 вход 32*32 ядро 5*5, сдвиг 1
• 2 – AlexNet вход 224*224 ядро 11*11, сдвиг 4, с дополнением
• 3 – VGG-19 вход 224*224 ядро 3*3, сдвиг 2, с дополнением
• 4 – ResNet вход 224*224 ядро 7*7, сдвиг 2, с дополнением
• 5 – SRCNN вход 32*32 ядро 9*9, сдвиг 1, с дополнением
Можно задавать несколько вариантов свёрток.
Например: «-f=145» означает: выбор 1,4,5 свёртки из списка с типом по умолчанию (если он не задан отдельным параметром).
Флаги для задания свёрток вручную
-rH*W, -tensorH*W – размерность входного тензора. Где H – высота, W – ширина. H и W могут быть заданы в диапазоне от 16 до 512.
Если указывается одно значение, например, -r448 – это означает, что H и W = 448.
Данный параметр обязателен.
Данный параметр может присутствовать только в одном экземпляре.
-kH*W -kernelH*W – размерность ядра. Где H – высота, W – ширина. H и W могут быть заданы в диапазоне от 3 до 31. По умолчанию используется H и W = 3.
Если указывается одно значение, например, -k5 – это означает, что H и W = 5.
Данный параметр может присутствовать только в одном экземпляре.
-sH*V, -strideH*V – сдвиг (stride) по горизонтали и вертикали. Если указано одно число, то оно применяется на две оси. N – от 1 (по умолчанию) до 512. Сдвиг уменьшается автоматически по размеру тензора).
Если указывается одно значение, например, -s10 – это означает, что H и V = 10.
Данный параметр может присутствовать только в одном экземпляре.
-pH*V, -paddingH*V – дополнение нулями (padding). Может быть 0 (нет дополнения) или 1 (есть дополнение). Если указана одна цифра, то она применяется на две оси. По умолчанию сдвига нет.
Если указывается одно значение, например, -p1 – это означает, что H и V = 1.
Данный параметр может присутствовать только в одном экземпляре.
-iZ, -inputsZ – число входных каналов (Z). Может быть от 1 до 256.
Данный параметр может присутствовать только в одном экземпляре.
-oZ, -outputsZ – число выходов (Z). Может быть от 1 до 256.
Данный параметр может присутствовать только в одном экземпляре.
-cR,K,S,P,I,O,T,Z -convR,K,S,P,I,O,T,Z - выбор используемых свёрток (одно или несколько значений). Где:
• R - H*W – размерность входного тензора;
• K - H*W – размерность ядра;
• S - H*V – сдвиг (stride);
• P - H*V – дополнение нулями (padding);
• I – число входных каналов;
• O – число выходов;
• T – используемые типы данных;
• Z – оптимизация.
При этом обязательные параметры это размер тензора и ядра, остальные можно не указывать (будут использованы значения по умолчанию).

Данный параметр может присутствовать в нескольких экземплярах, т.е.
можно задать несколько различных свёрток. Например: «-c224*224,3*3,1*1,1 -c256,11,2,0,1,64,f32i8,0125»

Таким образом, запуск программы на преднастроенных свёртках может выглядеть так:
./convbench -f=12345 -t=i8 -z=0125

Таким образом, запуск программы на заданных вручную свёртках может выглядеть так:
./convbench -r256 -k11 -s2 -p0 -i1 -o64 -t=f32i8 -z=0125
или
./convbench -c256,11,2,0,1,64,f32i8,0125

Пример конфигурационного json-файла:

```
{
   "config":{
      "quiet" : false,
      "log" : "newlog.txt",
      "verify" : 0.05
   },
   "conv":[
      {
         "tensor": "5*5",
         "kernel": "3*3",
         "stride": "1",
         "padding": "1",
         "inputs": "1",
         "types": "f32",
         "note": "тестовая свёртка"
      },
      {
         "tensor": "224*224",
         "kernel": "3*3",
         "stride": "2*2",
         "inputs": "3",
         "types": "i32"
      },
      {
         "tensor": "224*224",
         "kernel": "3*3",
         "stride": "1*1",
         "outputs": "16"
      }
   ]
}
```

В разделе «**config**»:

- «quiet» - тихий режим, аналогично ключу «**-q**». Может принимать значение false и true.
- «log» - признак логгирования в файл, аналогично ключу «**-l**». Значение соответствует имени лог-файла.
- «verify» - признак тестирования корректности свёртки, аналогично ключу «**-v**»

В разделе «**conv**» задаётся массив свёрток. Для каждой свёртки используются поля "tensor", "kernel", "stride", "padding", "inputs", "outputs", "types" — соответсвуют аналогичным ключам командной строки. При отсутствии полей берутся их значения по умолчанию.
Поле "note" задаёт текстовое описание указанной свёртки (выводится при расчёте).

Для проверки использования библиотеки приводится пример реализации выполнения математических операций с использованием доступных процессорных операций (SSE, AVX, AVX512).

Для этого:

1) Перейти в каталог example
cd ../example

2) Произвести сборку примера
mkdir build
cd build
cmake ..
make

3) В каталоге build появится исполняемый файл vecmulsum

4) Для запуска примера необходимо выполнить команду
./vecmulsum -t=i8 -i=../random_int8_1.bin -i=../random_int8_2.bin

vecmulsum имеет следующие флаги:

-h, --help, -? - вызов данной справки с последующим выходом.
-q, -quiet – тихий режим. Выводится только результат расчёта.
-t=Z, -type=Z - выбор обсчитываемого типа данных. Допустимые типы данных:
• f64 – double (знаковое с плавающей точкой 64 bit),
• f32 – float (знаковое с плавающей точкой 32 bit),
• i32 – int (знаковое целое 32 bit),
• i8 – char (знаковое целое 8 bit).
-i=filename, -infile=filename – выбор входных файлов (filename).
Должны быть указаны два имени (в двух параметрах) бинарных файлов, в каждом из которых сохранены данные вектора нужного типа.

В бинарных файлах примеров сохранены последовательности под разные типы данных и их необходимо использовать только с соответствующими ключами, описывающими этот тип данных. Например файлы "random_float*" используются только с ключом -t=f32, а файлы "random_int8*" только с ключом -t=i8