ggml(Georgi Gerganov's machine learning)

Roadmap / Manifesto

Tensor library for machine learning

Note that this project is under active development.
Some of the development is currently happening in the llama.cpp and whisper.cpp repos

Features

- Written in C
- 16-bit float support
- Integer quantization support (4-bit, 5-bit, 8-bit, etc.)
- Automatic differentiation
- ADAM and L-BFGS optimizers
- Optimized for Apple Silicon
- On x86 architectures utilizes AVX / AVX2 intrinsics
- On ppc64 architectures utilizes VSX intrinsics
- No third-party dependencies
- Zero memory allocations during runtime

Updates

- [x] Example of GPT-2 inference examples/gpt-2
- [x] Example of GPT-J inference examples/gpt-j
- [x] Example of Whisper inference ggerganov/whisper.cpp
- [x] Example of LLaMA inference ggerganov/llama.cpp
- [x] Example of LLaMA training ggerganov/llama.cpp/examples/baby-llama
- [x] Example of Falcon inference cmp-nct/ggllm.cpp
- [x] Example of BLOOM inference NouamaneTazi/bloomz.cpp
- [x] Example of RWKV inference saharNooby/rwkv.cpp
- [x] Example of SAM inference examples/sam
- [x] Example of BERT inference skeskinen/bert.cpp
- [x] Example of BioGPT inference PABannier/biogpt.cpp
- [x] Example of Encodec inference PABannier/encodec.cpp
- [x] Example of CLIP inference monatis/clip.cpp
- [x] Example of MiniGPT4 inference Maknee/minigpt4.cpp
- [x] Example of ChatGLM inference li-plus/chatglm.cpp
- [x] Example of Stable Diffusion inference leejet/stable-diffusion.cpp
- [x] Example of Qwen inference QwenLM/qwen.cpp
- [x] Example of YOLO inference examples/yolo
- [x] Example of ViT inference staghado/vit.cpp
- [x] Example of multiple LLMs inference foldl/chatllm.cpp
- [x] SeamlessM4T inference *(in development)*
  https://github.com/facebookresearch/seamless_communication/tree/main/ggml

Python environment setup and building the examples

```
git clone https://github.com/ggerganov/ggml
cd ggml
# Install python dependencies in a virtual environment
python3.10 -m venv ggml_env
source ./ggml_env/bin/activate
pip install -r requirements.txt
# Build the examples
mkdir build && cd build
cmake ..
cmake --build . --config Release -j 8
```

GPT inference (example)

With ggml you can efficiently run GPT-2 and GPT-J inference on the CPU.

Here is how to run the example programs:

```
# Run the GPT-2 small 117M model
../examples/gpt-2/download-ggml-model.sh 117M
./bin/gpt-2-backend -m models/gpt-2-117M/ggml-model.bin -p "This is an example"

# Run the GPT-J 6B model (requires 12GB disk space and 16GB CPU RAM)
../examples/gpt-j/download-ggml-model.sh 6B
./bin/gpt-j -m models/gpt-j-6B/ggml-model.bin -p "This is an example"

# Run the Cerebras-GPT 111M model
# Download from: https://huggingface.co/cerebras
python3 ../examples/gpt-2/convert-cerebras-to-ggml.py /path/to/Cerebras-GPT-111M/
./bin/gpt-2 -m /path/to/Cerebras-GPT-111M/ggml-model-f16.bin -p "This is an
example"
```

The inference speeds that I get for the different models on my 32GB MacBook M1 Pro are as follows:

Model	Size	Time / Token
GPT-2	117M	5 ms
GPT-2	345M	12 ms
GPT-2	774M	23 ms
GPT-2	1558M	42 ms
---	---	---
GPT-J	6B	125 ms

For more information, checkout the corresponding programs in the examples folder.

Using Metal (only with GPT-2)

For GPT-2 models, offloading to GPU is possible. Note that it will not improve inference performances but will reduce power consumption and free up the CPU for other tasks.

To enable GPU offloading on MacOS:

```
cmake -DGGML_METAL=ON -DBUILD_SHARED_LIBS=Off ..

# add -ngl 1
./bin/gpt-2 -t 4 -ngl 100 -m models/gpt-2-117M/ggml-model.bin -p "This is an
example"
```

Using cuBLAS

```
# fix the path to point to your CUDA compiler
cmake -DGGML_CUDA=ON -DCMAKE_CUDA_COMPILER=/usr/local/cuda-12.1/bin/nvcc ..
```

Using hipBLAS

```
cmake -DCMAKE_C_COMPILER="$(hipconfig -l)/clang" -DCMAKE_CXX_COMPILER="$(hipconfig
-l)/clang++" -DGGML_HIPBLAS=ON
```

Using SYCL

```
# linux
source /opt/intel/oneapi/setvars.sh
cmake -G "Ninja" -DCMAKE_C_COMPILER=icx -DCMAKE_CXX_COMPILER=icpx -DGGML_SYCL=ON
..

# windows
"C:\Program Files (x86)\Intel\oneAPI\setvars.bat"
cmake -G "Ninja" -DCMAKE_C_COMPILER=cl -DCMAKE_CXX_COMPILER=icx -DGGML_SYCL=ON ..
```

Compiling for Android

Download and unzip the NDK from this download page. Set the NDK_ROOT_PATH environment variable or provide the absolute path to the CMAKE_ANDROID_NDK in the command below.

```
cmake .. \
    -DCMAKE_SYSTEM_NAME=Android \
    -DCMAKE_SYSTEM_VERSION=33 \
```

```
    -DCMAKE_ANDROID_ARCH_ABI=arm64-v8a \
    -DCMAKE_ANDROID_NDK=$NDK_ROOT_PATH
    -DCMAKE_ANDROID_STL_TYPE=c++_shared
```

```
# Create directories
adb shell 'mkdir /data/local/tmp/bin'
adb shell 'mkdir /data/local/tmp/models'

# Push the compiled binaries to the folder
adb push bin/* /data/local/tmp/bin/

# Push the ggml library
adb push src/libggml.so /data/local/tmp/

# Push model files
adb push models/gpt-2-117M/ggml-model.bin /data/local/tmp/models/


# Now lets do some inference ...
adb shell

# Now we are in shell
cd /data/local/tmp
export LD_LIBRARY_PATH=/data/local/tmp
./bin/gpt-2-backend -m models/ggml-model.bin -p "this is an example"
```

Resources

- GGML - Large Language Models for Everyone: a description of the GGML format provided by the maintainers of the `llm` Rust crate, which provides Rust bindings for GGML
- marella/ctransformers: Python bindings for GGML models.
- go-skynet/go-ggml-transformers.cpp: Golang bindings for GGML models
- smspillaz/ggml-gobject: GObject-introspectable wrapper for use of GGML on the GNOME platform.