gemmafinetune-teamsecret-datanexus

July 9, 2024

1 Gemma 2b - Finetuning Attempt

[1]:
```
!pip3 install -q -U bitsandbytes==0.42.0
!pip3 install -q -U peft==0.8.2
!pip3 install -q -U trl==0.7.10
!pip3 install -q -U accelerate==0.27.1
!pip3 install -q -U datasets==2.17.0
!pip3 install -q -U transformers==4.38.0
```

```
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
kfp 2.5.0 requires google-cloud-storage<3,>=2.2.1, but you have google-cloud-
storage 1.44.0 which is incompatible.
ERROR: pip's dependency resolver does not currently take into account
all the packages that are installed. This behaviour is the source of the
following dependency conflicts.
cudf 24.4.1 requires cubinlinker, which is not installed.
cudf 24.4.1 requires cupy-cuda11x>=12.0.0, which is not installed.
cudf 24.4.1 requires ptxcompiler, which is not installed.
cuml 24.4.0 requires cupy-cuda11x>=12.0.0, which is not installed.
dask-cudf 24.4.1 requires cupy-cuda11x>=12.0.0, which is not installed.
cudf 24.4.1 requires cuda-python<12.0a0,>=11.7.1, but you have cuda-python
12.5.0 which is incompatible.
distributed 2024.1.1 requires dask==2024.1.1, but you have dask 2024.5.2 which
is incompatible.
gcsfs 2024.3.1 requires fsspec==2024.3.1, but you have fsspec 2023.10.0 which is
incompatible.
rapids-dask-dependency 24.4.1a0 requires dask==2024.1.1, but you have dask
2024.5.2 which is incompatible.
rapids-dask-dependency 24.4.1a0 requires dask-expr==0.4.0, but you have dask-
expr 1.1.2 which is incompatible.
s3fs 2024.3.1 requires fsspec==2024.3.1, but you have fsspec 2023.10.0 which is
incompatible.
```

[2]:
```python
# import os
# from google.colab import userdata
# os.environ["HF_TOKEN"] = userdata.get('HF_TOKEN')
```

[3]:
```python
from huggingface_hub import notebook_login
notebook_login()
```

```
VBox(children=(HTML(value='<center> <img\nsrc=https://huggingface.co/front/
↪assets/huggingface_logo-noborder.sv…
```

[4]:
```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM, BitsAndBytesConfig

bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.bfloat16
)
```

[5]:
```
model_id = "google/gemma-2b-it"
model = AutoModelForCausalLM.from_pretrained(model_id,␣
↪quantization_config=bnb_config, device_map={"":0})
tokenizer = AutoTokenizer.from_pretrained(model_id, add_eos_token=True)
```

```
/opt/conda/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132:
FutureWarning: `resume_download` is deprecated and will be removed in version
1.0.0. Downloads always resume when possible. If you want to force a new
download, use `force_download=True`.
  warnings.warn(

config.json:   0%|          | 0.00/627 [00:00<?, ?B/s]

/opt/conda/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132:
FutureWarning: `resume_download` is deprecated and will be removed in version
1.0.0. Downloads always resume when possible. If you want to force a new
download, use `force_download=True`.
  warnings.warn(

model.safetensors.index.json:   0%|          | 0.00/13.5k [00:00<?, ?B/s]

Downloading shards:   0%|          | 0/2 [00:00<?, ?it/s]

model-00001-of-00002.safetensors:   0%|          | 0.00/4.95G [00:00<?, ?B/s]

model-00002-of-00002.safetensors:   0%|          | 0.00/67.1M [00:00<?, ?B/s]

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

generation_config.json:   0%|          | 0.00/137 [00:00<?, ?B/s]

tokenizer_config.json:   0%|          | 0.00/34.2k [00:00<?, ?B/s]

tokenizer.model:   0%|          | 0.00/4.24M [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/17.5M [00:00<?, ?B/s]

special_tokens_map.json:   0%|          | 0.00/636 [00:00<?, ?B/s]
```

[5]:
```
from datasets import load_dataset

dataset = load_dataset('csv', data_files='/kaggle/input/teamsecret/
 ↪finetune_training_data.csv')

dataset
```

```
Generating train split: 0 examples [00:00, ? examples/s]
```

[5]:
```
DatasetDict({
    train: Dataset({
        features: ['ocr', 'output'],
        num_rows: 97
    })
})
```

[6]:
```
dataset = dataset['train']
```

[7]:
```
def generate_prompt(data_point):

    prefix_text = 'Below is an OCR text that describes a invoice OCR scan.␣
 ↪Write a json response that ' \
                'appropriately completes the request.\n\n'
    # Samples with additional context into.

    text = f"""<start_of_turn>user {prefix_text} here are the inputs␣
 ↪{data_point["ocr"]}␣
 ↪<end_of_turn>\n<start_of_turn>model{data_point["output"]} <end_of_turn>"""

    return text

text_column = [generate_prompt(data_point) for data_point in dataset]
dataset = dataset.add_column("prompt", text_column)
```

[8]:
```
dataset
```

[8]:
```
Dataset({
    features: ['ocr', 'output', 'prompt'],
    num_rows: 97
})
```

[9]:
```
dataset = dataset.shuffle(seed=1234)  # Shuffle dataset here
dataset = dataset.map(lambda samples: tokenizer(samples["prompt"]),␣
 ↪batched=True)
```

```
Map:   0%|          | 0/97 [00:00<?, ? examples/s]
```

[10]:
```
dataset = dataset.train_test_split(test_size=0.25)
train_data = dataset['train']
test_data = dataset['test']
```

[11]:
```
print(test_data)
```

```
Dataset({
    features: ['ocr', 'output', 'prompt', 'input_ids', 'attention_mask'],
    num_rows: 25
})
```

[13]:
```
from peft import LoraConfig, PeftModel, prepare_model_for_kbit_training,␣
 ↪get_peft_model
model.gradient_checkpointing_enable()
model = prepare_model_for_kbit_training(model)
```

[14]:
```
model
```

[14]:
```
GemmaForCausalLM(
  (model): GemmaModel(
    (embed_tokens): Embedding(256000, 2048, padding_idx=0)
    (layers): ModuleList(
      (0-17): 18 x GemmaDecoderLayer(
        (self_attn): GemmaSdpaAttention(
          (q_proj): Linear4bit(in_features=2048, out_features=2048, bias=False)
          (k_proj): Linear4bit(in_features=2048, out_features=256, bias=False)
          (v_proj): Linear4bit(in_features=2048, out_features=256, bias=False)
          (o_proj): Linear4bit(in_features=2048, out_features=2048, bias=False)
          (rotary_emb): GemmaRotaryEmbedding()
        )
        (mlp): GemmaMLP(
          (gate_proj): Linear4bit(in_features=2048, out_features=16384,
bias=False)
          (up_proj): Linear4bit(in_features=2048, out_features=16384,
bias=False)
          (down_proj): Linear4bit(in_features=16384, out_features=2048,
bias=False)
          (act_fn): GELUActivation()
        )
        (input_layernorm): GemmaRMSNorm()
        (post_attention_layernorm): GemmaRMSNorm()
      )
    )
    (norm): GemmaRMSNorm()
  )
  (lm_head): Linear(in_features=2048, out_features=256000, bias=False)
)
```

[15]:
```
import bitsandbytes as bnb
def find_all_linear_names(model):
    cls = bnb.nn.Linear4bit #if args.bits == 4 else (bnb.nn.Linear8bitLt if␣
 ↪args.bits == 8 else torch.nn.Linear)
    lora_module_names = set()
    for name, module in model.named_modules():
    if isinstance(module, cls):
        names = name.split('.')
        lora_module_names.add(names[0] if len(names) == 1 else names[-1])
    if 'lm_head' in lora_module_names: # needed for 16-bit
        lora_module_names.remove('lm_head')
    return list(lora_module_names)
```

[16]:
```
modules = find_all_linear_names(model)
print(modules)
```

```
['down_proj', 'k_proj', 'up_proj', 'v_proj', 'gate_proj', 'q_proj', 'o_proj']
```

[17]:
```
from peft import LoraConfig, get_peft_model

lora_config = LoraConfig(
    r=64, # 2*lora_alpha
    lora_alpha=32,
    target_modules=modules,
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
)

model = get_peft_model(model, lora_config)
```

[18]:
```
trainable, total = model.get_nb_trainable_parameters()
print(f"Trainable: {trainable} | total: {total} | Percentage: {trainable/
 ↪total*100:.4f}%")
```

```
Trainable: 78446592 | total: 2584619008 | Percentage: 3.0351%
```

[21]:
```
#new code using SFTTrainer
import transformers

from trl import SFTTrainer

tokenizer.pad_token = tokenizer.eos_token
torch.cuda.empty_cache()

trainer = SFTTrainer(
    model=model,
```

```
    train_dataset=train_data,
    eval_dataset=test_data,
    dataset_text_field="prompt",
    peft_config=lora_config,
    args=transformers.TrainingArguments(
        per_device_train_batch_size=1,
        gradient_accumulation_steps=4,
        warmup_steps=0.03,
        max_steps=20,
        learning_rate=2e-4,
        logging_steps=1,
        output_dir="/kaggle/working/outputs2",
        optim="paged_adamw_8bit",
        save_strategy="epoch",
    ),
    data_collator=transformers.DataCollatorForLanguageModeling(tokenizer,␣
  ↪mlm=False),
)
```

[22]:
```
model.config.use_cache = False  # silence the warnings. Please re-enable for␣
  ↪inference!
trainer.train()
```

```
wandb: Logging into wandb.ai. (Learn how to deploy a W&B server
locally: https://wandb.me/wandb-server)
wandb: You can find your API key in your browser here:
https://wandb.ai/authorize
wandb: Paste an API key from your profile and hit enter, or press
ctrl+c to quit:
  ····································

wandb: Appending key for api.wandb.ai to your netrc file:
/root/.netrc

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

/opt/conda/lib/python3.10/site-packages/torch/utils/checkpoint.py:429:
UserWarning: torch.utils.checkpoint: please pass in use_reentrant=True or
use_reentrant=False explicitly. The default value of use_reentrant will be
updated to be False in the future. To maintain current behavior, pass
use_reentrant=True. It is recommended that you use use_reentrant=False. Refer to
```

```
docs for more details on the differences between the two variants.
  warnings.warn(
```

```
<IPython.core.display.HTML object>
```

```
/opt/conda/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132:
FutureWarning: `resume_download` is deprecated and will be removed in version
1.0.0. Downloads always resume when possible. If you want to force a new
download, use `force_download=True`.
  warnings.warn(
/opt/conda/lib/python3.10/site-packages/torch/utils/checkpoint.py:429:
UserWarning: torch.utils.checkpoint: please pass in use_reentrant=True or
use_reentrant=False explicitly. The default value of use_reentrant will be
updated to be False in the future. To maintain current behavior, pass
use_reentrant=True. It is recommended that you use use_reentrant=False. Refer to
docs for more details on the differences between the two variants.
  warnings.warn(
/opt/conda/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132:
FutureWarning: `resume_download` is deprecated and will be removed in version
1.0.0. Downloads always resume when possible. If you want to force a new
download, use `force_download=True`.
  warnings.warn(
```

[22]:
```
TrainOutput(global_step=20, training_loss=1.4077922224998474,
metrics={'train_runtime': 256.8044, 'train_samples_per_second': 0.312,
'train_steps_per_second': 0.078, 'total_flos': 951922373640192.0, 'train_loss':
1.4077922224998474, 'epoch': 1.11})
```

[23]:
```
new_model = "/kaggle/working/AnuSangha_GemmaFineTuned2"
```

[24]:
```
trainer.model.save_pretrained(new_model)
```

```
/opt/conda/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132:
FutureWarning: `resume_download` is deprecated and will be removed in version
1.0.0. Downloads always resume when possible. If you want to force a new
download, use `force_download=True`.
  warnings.warn(
```

[25]:
```
model_id = 'google/gemma-2b-it'
```

[26]:
```
base_model = AutoModelForCausalLM.from_pretrained(
    model_id,
    low_cpu_mem_usage=True,
    return_dict=True,
    torch_dtype=torch.float16,
    device_map={"": 0},
    # force_download=True
)
```

```
merged_model= PeftModel.from_pretrained(base_model, new_model)
merged_model= merged_model.merge_and_unload()

# # Save the merged model
merged_model.save_pretrained("/kaggle/working/
 ↪AnuSangha_GemmaFineTuned_FINAL2",safe_serialization=True)
tokenizer.save_pretrained("/kaggle/working/AnuSangha_GemmaFineTuned_FINAL2")
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right"
```

```
Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]
```

[27]:
```
merged_model.push_to_hub('anuraaga/Gemma-dn3')
tokenizer.push_to_hub('anuraaga/Gemma-dn3')
```

```
model-00001-of-00002.safetensors:   0%|          | 0.00/4.95G [00:00<?, ?B/s]

Upload 2 LFS files:   0%|          | 0/2 [00:00<?, ?it/s]

model-00002-of-00002.safetensors:   0%|          | 0.00/67.1M [00:00<?, ?B/s]

README.md:   0%|          | 0.00/5.17k [00:00<?, ?B/s]

Upload 2 LFS files:   0%|          | 0/2 [00:00<?, ?it/s]

tokenizer.model:   0%|          | 0.00/4.24M [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/17.5M [00:00<?, ?B/s]
```

[27]:
```
CommitInfo(commit_url='https://huggingface.co/anuraaga/gemma-
dn3/commit/a39cba92c51c487e9da32bd15e9d7b7625b37e76', commit_message='Upload
tokenizer', commit_description='',
oid='a39cba92c51c487e9da32bd15e9d7b7625b37e76', pr_url=None, pr_revision=None,
pr_num=None)
```

2 test

[16]:
```
def generate_prompt(i):
    prefix_text = 'Below is an OCR text that describes an invoice OCR scan.␣
 ↪Write a json response that appropriately completes the request.\n\n'
    text = f"""<start_of_turn>user {prefix_text} here are the inputs␣
 ↪{inputs[i]} <end_of_turn>\n<start_of_turn>model <end_of_turn>"""
    return text
```

[2]:
```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM
model_id = "anuraaga/Gemma-dn3"
model = AutoModelForCausalLM.from_pretrained(model_id,  device_map={"":0})
tokenizer = AutoTokenizer.from_pretrained(model_id, add_eos_token=True)
```

```
config.json:   0%|          | 0.00/662 [00:00<?, ?B/s]
model.safetensors.index.json:   0%|          | 0.00/13.5k [00:00<?, ?B/s]
Downloading shards:   0%|          | 0/2 [00:00<?, ?it/s]
model-00001-of-00002.safetensors:   0%|          | 0.00/4.95G [00:00<?, ?B/s]
model-00002-of-00002.safetensors:   0%|          | 0.00/67.1M [00:00<?, ?B/s]
`config.hidden_act` is ignored, you should use `config.hidden_activation`
instead.
Gemma's activation function will be set to `gelu_pytorch_tanh`. Please, use
`config.hidden_activation` if you want to override this behaviour.
See https://github.com/huggingface/transformers/pull/29402 for more details.
Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]
generation_config.json:   0%|          | 0.00/132 [00:00<?, ?B/s]
tokenizer_config.json:   0%|          | 0.00/40.6k [00:00<?, ?B/s]
tokenizer.model:   0%|          | 0.00/4.24M [00:00<?, ?B/s]
tokenizer.json:   0%|          | 0.00/17.5M [00:00<?, ?B/s]
special_tokens_map.json:   0%|          | 0.00/522 [00:00<?, ?B/s]
```

[12]:
```
inputs = test_data['ocr']
outputs = test_data['output']
```

[28]:
```
query = inputs[0]
```

[14]:
```
outputs[0]
```

[14]:
```
"{'client': None, 'client_tax_id': None, 'header': {'client': 'Coleman Inc 44067
Woods Meadows Suite 659 Mariamouth, UT 44968', 'client_tax_id': '918-88-4124',
'iban': 'GB46MLEX00282137220677', 'invoice_date': '08/03/2013', 'invoice_no':
'20653012', 'seller': 'Smith and Sons 237 Steven Views Lake Robert, MI 28151',
'seller_tax_id': '912-86-5998'}, 'iban': None, 'invoice_date': None,
'invoice_no': None, 'item_desc': None, 'item_gross_worth': None,
'item_net_price': None, 'item_net_worth': None, 'item_qty': None, 'item_vat':
None, 'items': array([{'iban': None, 'item_desc': 'Sony PlayStation 1 PS1
Console model scph-7501', 'item_gross_worth': '16.49', 'item_net_price':
'14.99', 'item_net_worth': '14.99', 'item_qty': '1.00', 'item_vat': '10%',
'total_net_worth': None},\n       {'iban': None, 'item_desc': 'Nintendo 64
Console with Box, Foam and extra oem controller', 'item_gross_worth': '528.00',
'item_net_price': '120.00', 'item_net_worth': '480.00', 'item_qty': '4.00',
'item_vat': '10%', 'total_net_worth': None},\n       {'iban': None, 'item_desc':
'B311 Nintendo 3DS console Cosmo Black Japan X', 'item_gross_worth': '52.76',
'item_net_price': '11.99', 'item_net_worth': '47.96', 'item_qty': '4.00',
'item_vat': '10%', 'total_net_worth': None},\n       {'iban': None, 'item_desc':
'PocketGo Bittboy v2 White Retro Video Game Portable Console - 3.5 IPS display',
```

```
'item_gross_worth': '42.90', 'item_net_price': '39.00', 'item_net_worth':
'39.00', 'item_qty': '1.00', 'item_vat': '10%', 'total_net_worth': None}],\n
dtype=object), 'seller': None, 'seller_tax_id': None, 'summary':
{'total_gross_worth': '$640.15', 'total_net_worth': '$581.95', 'total_vat':
'$58.20'}, 'total_gross_worth': None, 'total_net_worth': None, 'total_vat':
None}"
```

[17]:
```
prompt = generate_prompt(0)
prompt
```

[17]:
```
'<start_of_turn>user Below is an OCR text that describes an invoice OCR scan.
Write a json response that appropriately completes the request.\n\n here are the
inputs Invoice no: 20653012\n\nDate of issue:\n\nSeller:\n\nSmith and Sons\n237
Steven Views\nLake Robert, MI 28151\n\nTax Id: 912-86-5998\nIBAN:
GB46MLEX00282137220677\n\nITEMS\nNo. Description Qty\n1. Sony PlayStation 1 PS1
Console 1,00\n\nmodel scph-7501\n\n2. Nintendo 64 Console with Box, 4,00\nFoam
and extra oem controller\n\n3. B311 Nintendo 3DS console 4,00\nCosmo Black Japan
x\n\n4. PocketGo Bittboy v2 White 1,00\nRetro Video Game Portable\nConsole - 3.5
IPS display\n\nSUMMARY\n\nVAT [%]\n10%\n\nTotal\n\n08/03/2013\n\nUM\n\neach\n\ne
ach\n\neach\n\neach\n\nClient:\n\nColeman Inc\n44067 Woods Meadows Suite
659\nMariamouth, UT 44968\n\nTax Id: 918-88-4124\n\nNet price Net worth VAT
[%]\n\n14,99 14,99 10%\n120,00 480,00 10%\n11,99 47,96 10%\n39,00 39,00 10%\nNet
worth VAT\n581,95 58,20\n$ 581,95 $
58,20\n\nGross\n\nworth\n\n16,49\n\n528,00\n\n52,76\n\n42,90\n\nGross
worth\n640,15\n\n$ 640,15\n\x0c <end_of_turn>\n<start_of_turn>model
<end_of_turn>'
```

[18]:
```
encodeds = tokenizer(prompt, return_tensors="pt", add_special_tokens=True)
```

[19]:
```
model_inputs = encodeds.to(device = "cuda:0")
```

[20]:
```
generated_ids = model.generate(**model_inputs, max_new_tokens=1000,␣
↪do_sample=True, pad_token_id=tokenizer.eos_token_id)
```

[23]:
```
len(generated_ids[0])
```

[23]:
```
639
```

[24]:
```
# decoded = tokenizer.batch_decode(generated_ids)
decoded = tokenizer.decode(generated_ids[0], skip_special_tokens=True)
```

[25]:
```
decoded
```

[25]:
```
'user Below is an OCR text that describes an invoice OCR scan. Write a json
response that appropriately completes the request.\n\n here are the inputs
Invoice no: 20653012\n\nDate of issue:\n\nSeller:\n\nSmith and Sons\n237 Steven
Views\nLake Robert, MI 28151\n\nTax Id: 912-86-5998\nIBAN:
```

```
GB46MLEX00282137220677\n\nITEMS\nNo. Description Qty\n1. Sony PlayStation 1 PS1
Console 1,00\n\nmodel scph-7501\n\n2. Nintendo 64 Console with Box, 4,00\nFoam
and extra oem controller\n\n3. B311 Nintendo 3DS console 4,00\nCosmo Black Japan
x\n\n4. PocketGo Bittboy v2 White 1,00\nRetro Video Game Portable\nConsole - 3.5
IPS display\n\nSUMMARY\n\nVAT [%]\n10%\n\nTotal\n\n08/03/2013\n\nUM\n\neach\n\ne
ach\n\neach\n\neach\n\nClient:\n\nColeman Inc\n44067 Woods Meadows Suite
659\nMariamouth, UT 44968\n\nTax Id: 918-88-4124\n\nNet price Net worth VAT
[%]\n\n14,99 14,99 10%\n120,00 480,00 10%\n11,99 47,96 10%\n39,00 39,00 10%\nNet
worth VAT\n581,95 58,20\n$ 581,95 $
58,20\n\nGross\n\nworth\n\n16,49\n\n528,00\n\n52,76\n\n42,90\n\nGross
worth\n640,15\n\n$ 640,15\n\x0c \nmodel
,2624,00\n\n,0000\n\n,0000\n\n,0000\n\n,0000\n\n,0000\n\nClient:\nColeman
Inc\n44067 Woods Meadows Suite 659\nMariamouth, UT 44968\n\nNet price Net worth
VAT [%] \n14,99 14,99 10%\n120,00 480,00 10%\n11,99 47,96 10%\n39,00 39,00 10%
\nNet worth VAT Gross\n581,95 58,20 640,15\n$ 640,15'
```

[33]:
```python
def get_completion(query: str, model, tokenizer) -> str:
    device = "cuda:0"

    prompt_template = """
    <start_of_turn>user
    Below is an OCR text that describes a invoice OCR scan. Write a json␣
 ↪response that ' \
                'appropriately completes the request.\n\n
    {query}
    <end_of_turn>\n<start_of_turn>model \n\n\n\n


    """
    prompt = prompt_template.format(query=query)

    encodeds = tokenizer(prompt, return_tensors="pt", add_special_tokens=True)

    model_inputs = encodeds.to(device)


    generated_ids = model.generate(**model_inputs, max_new_tokens=1000,␣
 ↪do_sample=True, pad_token_id=tokenizer.eos_token_id)
    # decoded = tokenizer.batch_decode(generated_ids)
    decoded = tokenizer.decode(generated_ids[0], skip_special_tokens=True)
    return (decoded)
```

[35]:
```python
response = get_completion(query, model, tokenizer)
```

[45]:
```python
print(response)
```

```
    user
    Below is an OCR text that describes a invoice OCR scan. Write a json
response that '                'appropriately completes the request.


    Invoice no: 20653012

Date of issue:

Seller:

Smith and Sons
237 Steven Views
Lake Robert, MI 28151

Tax Id: 912-86-5998
IBAN: GB46MLEX00282137220677

ITEMS
No. Description Qty
1. Sony PlayStation 1 PS1 Console 1,00

model scph-7501

2. Nintendo 64 Console with Box, 4,00
Foam and extra oem controller

3. B311 Nintendo 3DS console 4,00
Cosmo Black Japan x

4. PocketGo Bittboy v2 White 1,00
Retro Video Game Portable
Console - 3.5 IPS display

SUMMARY

VAT [%]
10%

Total

08/03/2013

UM

each

each
```

each

each

Client:

Coleman Inc
44067 Woods Meadows Suite 659
Mariamouth, UT 44968

Tax Id: 918-88-4124

Net price Net worth VAT [%]

14,99 14,99 10%
120,00 480,00 10%
11,99 47,96 10%
39,00 39,00 10%
Net worth VAT
581,95 58,20
$ 581,95 $ 58,20

Gross

worth

16,49

528,00

52,76

42,90

Gross worth
640,15

$ 640,15

model

```
    {
        "@context": "https://schema.org/ ",
        "@type": "Invoice",
        "additionalData": {
            "@type": "PropertyValue",
            "net price": "14,99",
            "net worth": "14,99",
            "gross worth": "640,15",
            "iban": "GB46MLEX00282137220677",
            "invoiceDate": "08/03/2013",
            "item": [
                {
                    "@type": "Item",
                    "description": "Sony PlayStation 1 PS1 Console model
scph-7501",
                    "quantity": "1,00",
                    "unique": "20653012"
                },
                // Skip "item" objects for other items
            ],
            "seller": {
                "@type": "Organization",
                "name": "Smith and Sons",
                "taxID": "912-86-5998"
            },
            "summary": {
                "@type": "ValueRange",
                "min": "14,99",
                "max": "16,49"
            }
        },
        "client": {
            "@type": "Organization",
            "name": "Coleman Inc",
            "taxID": "918-88-4124"
        },
        "seller": {
            "@type": "Person",
            "firstName": "Smith",
            "lastName": "John"
        }
    }
```