Adapting Neural Networks to Fixed-Point and Integer Arithmetic

Hanane BENMAGHNIA

hanane.benmaghnia@etudiant.univ-perp.fr

Supervisors: Matthieu MARTEL and Yasmine SELADJI

Laboratory of Mathematics and Physics (LAMPS)

University of Perpignan Via Domitia

June 20, 2019
Summary

1 Introduction

2 Neural Networks

3 Fixed-Point and Integer Arithmetic

4 Contributions
   ● Operations between operands with the same format
   ● Operations between operands with different formats
   ● Format of weighted sum

5 Conclusion & Perspectives
Plan

1. Introduction
2. Neural Networks
3. Fixed-Point and Integer Arithmetic
4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum
5. Conclusion & Perspectives
Introduction

Difficulty to run tasks in real time

Embedded systems have often low memory

Difficulty maintaining efficiency with limited resources

New Problem:

Artificial Intelligence and especially Neural Networks are increasingly used in Embedded Systems !!!
Contribution

**Input:** A trained NN, working at some precision (Floating-Point Format)

**Output:** Code generation (the new NN) to "simulate" the original NN using integers and fixed-point arithmetic

**Correctness:** The new NN with smaller data types behaves almost like the original NN

- Function Approximation: results stay in a desired hull
- Classification: only x% decisions differ from original NN

"Simulate": Same behavior as the original network with x% error
Plan

1. Introduction
2. Neural Networks
3. Fixed-Point and Integer Arithmetic
4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum
5. Conclusion & Perspectives
Neural Networks (NN)

- The conception is inspired by the biological functioning of neurons
- NN create fast classifications and generalizations
- NN are used on a variety of tasks (object recognition, speech and handwriting recognition, machine translation, medical diagnosis...)

![Diagram of a neural network with input and output layers, and hidden layers.](image)

Hanane BENMAGHNIA
hanane.benmaghnia@etudiant.univ-perp.fr
Supervisors: Matthieu MARTEL and Yasmine SELADJI
Laboratory of Mathematics and Physics (LAMPS) University of Perpignan Via Domitia

IA
June 20, 2019 7/31
Plan

1. Introduction

2. Neural Networks

3. Fixed-Point and Integer Arithmetic

4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum

5. Conclusion & Perspectives
Fixed-Point Format (FPF)

Float numbers are represented by integers in a format \( <m, l> \) or \( <I, F> \)

**Definition**

For \( X = (x_{n-1}x_{n-2}...x_1x_0)\beta \) and \( x = X.\beta^{-l} \), we have: \( p = m + l + 1 \).

- \( x \): Fixed-Point number with implicit scale factor \( \beta^{-l} \),
- \( X \): Integer representation,
- \( p \): Word length,
- \( m \): The most significant bit,
- \( l \): The least significant bit

Representation of \( X \) in a format \( <m, l> = <2, 6> \)

- \( m = 2 \)
- \( l = 6 \)

⚠️ No exponent, the programmer must take care of the formats!
Different interpretations of the same integer representation

<table>
<thead>
<tr>
<th>Scaling Factor Symbol</th>
<th>Format</th>
<th>Value of X</th>
<th>Value of x</th>
</tr>
</thead>
<tbody>
<tr>
<td>purple</td>
<td>&lt; 2, 6 &gt;</td>
<td>(10011010)₂ = (154)₁₀</td>
<td>(10.011010)₂ = (2.40625)₁₀</td>
</tr>
<tr>
<td>cyan</td>
<td>&lt; 5, 3 &gt;</td>
<td>(10011010)₂ = (154)₁₀</td>
<td>(10011.010)₂ = (19.25)₁₀</td>
</tr>
<tr>
<td>blue</td>
<td>&lt; 9, -1 &gt;</td>
<td>(10011010)₂ = (154)₁₀</td>
<td>(100110100)₂ = (308)₁₀</td>
</tr>
</tbody>
</table>

Representation of 154 in different formats
Plan

1. Introduction

2. Neural Networks

3. Fixed-Point and Integer Arithmetic

4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum

5. Conclusion & Perspectives
Plan

1. Introduction
2. Neural Networks
3. Fixed-Point and Integer Arithmetic
4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum
5. Conclusion & Perspectives
Addition

Let: \( x_1 \) and \( x_2 \) FP numbers with FPF \(<i, f>\):

\[
x_1 + x_2 = x_3
\]

\[
<i, f> + <i, f> = <i, f>
\]  \hspace{1cm} (1)

\[
\begin{cases}
0 < i < p \\
0 < i + f < p \\
f = p - i
\end{cases}
\]  \hspace{1cm} (2)

Example:

\[
\begin{array}{c}
000000.010 \quad <6, 3> \\
+ \\
001000.000 \quad <6, 3>
\end{array}
\]

\[
\overline{001000.010} \quad <6, 3>
\]
Multiplication

Let: $x_1$ and $x_2$ FP numbers with FPF $<i, f>$:

$$x_1 \times x_2 = x_3$$

$$<i, f> \times <i, f> = <i - f, 2f> \quad (3)$$

$$\begin{cases} 
0 < i < p \\
0 < i + f < p \\
f = p - i
\end{cases} \quad (4)$$
\[ x_1 \gg k = x_2 \]

\[ < i, f > \gg k = < i + k, f - k > \] \hspace{1cm} (5)

\[
\begin{cases}
0 < i < p \\
0 < i + f < p \\
f = p - i
\end{cases}
\] \hspace{1cm} (6)
$x_1 \ll k = x_2$

\[ <i, f> \ll k = <i - k, f + k> \]  \quad \text{(7)}

\[
\begin{cases}
0 < i < p \\
0 < i + f < p \\
f = p - i
\end{cases}
\]  \quad \text{(8)}
x_1 \ AND \ x_2 = x_3

\begin{align*}
\langle i, f \rangle \ AND \ & \langle i, f \rangle = \langle i, f \rangle \\
\left\{ \\
& 0 < i < p \\
& 0 < i + f < p \\
& f = p - i
\right. 
\end{align*}

(9)  

Same format for OR and XOR!
Plan

1. Introduction
2. Neural Networks
3. Fixed-Point and Integer Arithmetic
4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum
5. Conclusion & Perspectives
Addition

\[ x_1 + x_2 = x_3 \]

\[ < i_1, f_1 > + < i_2, f_2 > = < i_3, f_3 > \] (11)

\[
\begin{cases}
0 < i_3 < p \\
0 < i_3 + f_3 < p \\
i_3 = \text{max}(i_1, i_2) \\
f_3 = p - i_3
\end{cases}
\] (12)

**Example 1: fixed format**

```
00000.01  <5,2>  \\
+        \\
1000.0001 <4,4>  \\
\hline
?        <6,3>
```
Example 1

\[
\begin{align*}
0.000001 & <5,2> \\
+ & \\
1.000001 & <4,4> \\
\hline
? & <6,3>
\end{align*}
\]

\[
\begin{align*}
0.000000.010 & <6,3> \\
+ & \\
0.010000.000 & <6,3> \\
\hline
0.001000.010 & <6,3>
\end{align*}
\]
Example 2: fixed precision

\[
\begin{array}{c}
0.00000.01 \\
+ \\
1000.0001 \\
\hline
\end{array}
\]

\[
\begin{array}{c}
<5,2> \\
<4,4> \\
p=9 \\
\hline
<?, ?> \\
\end{array}
\]
Example 2

\[
\begin{align*}
0.000001 & \quad <5,2> \\
+ & \\
1.000001 & \quad <4,4> \\
\hline & \quad ? & \quad ?
\end{align*}
\]

\[p = 9\]

\[
\begin{align*}
0.000001 & \quad <5,2> \\
+ & \\
1.000001 & \quad <4,4> \\
\hline & \quad 0.10000101 & \quad <5,4>
\end{align*}
\]
Multiplication

\[ x_1 \times x_2 = x_3 \]

\[ < i_1, f_1 > \times < i_2, f_2 > = < i_3, f_3 > \]  \hspace{1cm} (13)

\[
\begin{align*}
0 < i_3 &< p \\
0 < i_3 + f_3 &< p \\
i_3 &= i_1 + i_2 - 1 \\
f_3 &= p - i_3
\end{align*}
\]  \hspace{1cm} (14)

Example: fixed precision

\[
\begin{array}{c}
00000.01 \\
* \\
1000.0001 \\
\hline
00000010.000001 \\
\end{array}
\]

\[ <5,2> \]

\[ <4,4> \\
\]

\[ <8,6> \]

\[ = 2.0156 \]
Example

\[0.0000001 \times 1000.0001 = 0.0000010.000001 = 2.0156\]

error = 0.0156
AND, OR, XOR

\[ x_1 \text{ AND } x_2 = x_3 \]

\[ <i_1, f_1> \text{ AND } <i_2, f_2> = <i_3, f_3> \quad (15) \]

\[
\begin{cases}
0 < i_3 < p \\
0 < i_3 + f_3 < p \\
i_3 = \max(i_1, i_2) \\
f_3 = p - i_3
\end{cases}
\quad (16)\]

Same format for OR and XOR!
Plan

1. Introduction
2. Neural Networks
3. Fixed-Point and Integer Arithmetic
4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum
5. Conclusion & Perspectives
Format of weighted sum

Let:

\[ f(\hat{x}) = W\hat{x} + b \]

\[ f(\hat{x}) = \sum_{j=0}^{n} w_{ij} \hat{x}_j + b_i \]

\[ f(\hat{x}) = \begin{cases} \sum_{j=1}^{i_2} w_{ij} \hat{x}_j & \text{if } i_2 \text{ is even} \\ \sum_{j=0}^{i_2-1} w_{ij} \hat{x}_j & \text{if } i_2 \text{ is odd} \end{cases} + b_i \]

\[ <i_1, f_1> \times <i_1, f_1> + <i_2, f_2> \times <i_1, f_1> + <i_2, f_2> \times <i_1, f_1> + <i_3, f_3> \]

\[ <i_4, f_4> = <i_1 + i_2 - 1, f_1 + f_2> \]

\[ <i_5, f_5> = \max(i_1 + i_2 - 1, i_3), p - \max((i_1 + i_2 - 1)) \]

\[ <i_5, f_5> = \max((i_1 + i_2 - 1), i_3), p - \max((i_1 + i_2 - 1)) \]

(W: matrix of weights, \( \hat{x} \): inputs and \( b \): bias

\( <i_1, f_1> \): representation format of the input vector \( \hat{x} \)

\( <i_2, f_2> \): representation format of \( W \)

\( <i_3, f_3> \): representation format of bias \( b \)

\( <i_5, f_5> \): representation format of the result \( f(\hat{x}) \)
Plan

1. Introduction

2. Neural Networks

3. Fixed-Point and Integer Arithmetic

4. Contributions
   - Operations between operands with the same format
   - Operations between operands with different formats
   - Format of weighted sum

5. Conclusion & Perspectives
Conclusion & Perspectives

- Reducing the size and execution time of large NN with this approach
- Implementing a code transformation tool
- Experiments on large NN
- Handling classifiers and interpolators.
Related Work


Martel, Matthieu (2017). “Floating-Point Format Inference in Mixed-Precision”. In: NFM.

Gehr T. 2018 Singh G. 2019
Gehr T. 2018  Singh G. 2019
Martel 2017
Dutta S. 2018