Enzyme: High-Performance Automatic Differentiation of LLVM

William S. Moses (wmoses@mit.edu), Valentin Churavy (vchuravy@mit.edu)

MIT CSAIL

Computing derivatives is key to many machine learning algorithms. Existing approaches:

- **Differentiable DSLs** [TensorFlow] provide a new language where everything is differentiable. Must rewrite code.
- **Operator Overloading** [Adapt] [JAX] provide differentiable versions of existing language constructs. May require rewriting.
- **Source Rewriting** tools statically analyze code to produce a new gradient function in the source language.

This hinders application of ML to new domains!

### Existing Automatic Differentiation Tools

<table>
<thead>
<tr>
<th>Name</th>
<th>Overview</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>TensorFlow</td>
<td>System for large-scale machine learning</td>
<td>Julia, Swift, Rust, Nim</td>
</tr>
<tr>
<td>PyTorch</td>
<td>Allows differentiation of active variables in C++/CUDA</td>
<td>TensorFlow: A system for large-scale machine learning, in NIPS 2017 Workshop</td>
</tr>
<tr>
<td>JAX</td>
<td>Enables automatic differentiation on low-level programs</td>
<td>TensorFlow, Julia, Rust</td>
</tr>
<tr>
<td>Enzyme</td>
<td>Compiler plugin that performs reverse-mode automatic differentiation</td>
<td>LLVM, Fortran, LLVM IR</td>
</tr>
<tr>
<td>Tapenade</td>
<td>Performs automatic differentiation on low-level programs</td>
<td>LLVM, Fortran, LLVM IR</td>
</tr>
<tr>
<td>Adept</td>
<td>Performs automatic differentiation on low-level programs</td>
<td>LLVM, Fortran, LLVM IR</td>
</tr>
<tr>
<td>Ref</td>
<td>Performs automatic differentiation on low-level programs</td>
<td>LLVM, Fortran, LLVM IR</td>
</tr>
</tbody>
</table>

### Optimization and AD

All tools for existing code operate at the source level preventing optimizations before AD without reimplementing compiler analyzes and optimizations into the AD tool. While historically not considered necessary, we demonstrate in Figure 2 how crucial optimization prior to AD can be.

```c
// Approximate Einstein Field Equation
G_{\mu\nu} = \frac{\kappa}{2} R_{\mu\nu} - \frac{\kappa}{2} g_{\mu\nu} R + \kappa T_{\mu\nu}.
```

Figure 2. When differentiating `norm`, running LUCM prior to AD is asymptotically faster than running LUCM followed by LUCM.

### Usage

We provide Enzyme packages for PyTorch and TensorFlow that allow users to import foreign code into their ML workflow without rewriting.

```python
import torch
import enzyme

# Create some initial tensor
inputTensor = ...

# Apply foreign function
outputTensor = enzyme("test.c", "+f").apply(inputTensor)

# Derive gradient
outputTensor.backward()
```

Figure 3. Using Enzyme from PyTorch.

### Design

Conventional Wisdom: "AD is more effective in high-level compiled languages (e.g. Julia, Swift, Rust, Nim) than traditional ones such as C/C++, Fortran and LLVM IR […]" -Innes [6]

Enzyme turns over said wisdom by demonstrating successful and high-performance AD on low-level programs. By introducing new interprocedural analyses, Enzyme is able to extract all the required high-level semantics necessary to differentiate.

### Evaluation

Performing AD after optimization yields a 4.2x speedup over AD before optimization. This accounts for much, but not all, of Enzyme’s improvement over prior art (different cache and activity analysis implementations).

Figure 6. Relative speedup of AD systems on ADBench+ [8] benchmarks, higher is better. A red X denotes programs that an AD system does not produce a correct gradient. A value of 1.0 denotes the fastest system, whereas 0.5 denotes taking twice as long.

### Conclusion

Enzyme is compiler plugin that performs reverse-mode automatic differentiation of LLVM [7]. By performing AD after optimization, Enzyme is able to achieve state-of-the-art performance. It is easy to incorporate into existing tools and we have demonstrated taking derivatives of C/C++ via Clang, PyTorch [2], and TensorFlow [1] as well as Julia, Rust, and Swift. We’ve also demonstrated dynamic language support by using Enzyme to differentiate Julia [9].

For more information about installing and using Enzyme, please visit https://enzyme.mit.edu and come to our spotlight presentation!

---

**References & Acknowledgements**