

# TAINÉ ZHAO

✉ twshare@outlook.com · ☎ (+86) 176-0064-7398 · 🌐 thautwarm

## 🎓 EDUCATION

Nanjing University of Science and Technology, Jiangsu, China

09/15 – Current

Major: Mathematics and Applied Mathematics

## 👤 WORK EXPERIENCE

Microsoft Research Asia, Beijing, China

11/17 – 02/19

*Intern* Compiler Design, Distributed System, Machine Learning Framework

- Created an implementation of dynamically loading and updating distributed data schema for Microsoft GraphEngine
- Created prototypes for MSRA internal projects about future machine learning frameworks, including gluing Python with MySQL in runtime level, and an extended SQL
- Created a LLVM IR builder framework in .NET side, which is introduced into a future/extended version of GraphEngine to achieve JIT support
- Finished a part of lowering tasks for the first/basic language in the platform of that future/extended version of GraphEngine

## 🐱 PERSONAL PROJECTS

### YAPyPy

<https://github.com/Xython/YAPyPy>

Yet Another Python Python / Pure Python Compiler

- A pure Python compiler built on CPython, which provides full compatibility for multiple versions of CPython 3.x, currently support 3.6.x and 3.7.x
- Provided some syntax extensions like dictionary destructuring and pattern matching
- Capable of customizing parser and bytecode emitter, which enables making arbitrary syntactic/semantic extensions with CPython its own

### MLStyle.jl

<https://thautwarm.github.io/MLStyle.jl/latest/>

A Julia package that provides ML language infrastructures like extensible pattern matching, ADTs/GADTs, etc.

- Support a large number of patterns from other languages like Haskell, Elixir, F#, OCaml, etc.
- Provided a group of concise, intuitive and convenient interfaces to customize pattern matching (extensible pattern matching)
- Allow users to restrict the accessing of patterns via a customizable predicate in module level, to achieve a clean scoping and avoid conflicts against redefinitions of custom patterns
- Provided a homoiconic way to manipulation ASTs via pattern matching, which enables users to extract sub-patterns from given ASTs and rewrite them, without a prerequisite about Julia ASTs

### FSTan

<https://github.com/thautwarm/FSTan>

F# implementation of Lightweighted Higher Kinded Types and type classes

- Provided a set of commonly-used abstractions like Functor, Monad, MonadTrans, etc., and some commonly-used data types like State/StateT, Either/EitherT, etc.
- Support ad-hoc polymorphisms (via F#'s STRT), which is an advance comparing to other implementations
- Support separating data type definitions from interfacing/instancing type classes, which is an advance comparing to other implementations

### RSolve

<https://github.com/thautwarm/RSolve>

A general purposed solver for logic programming in Haskell

- Established an abstraction over unification algorithms, which could be applied to many concrete instances (introduced below)
- Provided some concrete instances like HM unification and option question puzzles
- No dependency but Haskell standard libraries

### LanguageCollections

<https://github.com/thautwarm/LanguageCollections>

List of languages invented by me, which has recorded my experience of learning this topic

## ⚙️ SKILLS

---

### Compiler - Front End

- Capable of creating LL(1) Parser Generator, understand LL(k), experienced in creating lexer generators and Parser Combinators . Have several implementations in Python and F#, one of which supports processing context sensitive cases and has some demos about implementing parsers for languages that look like Haskell or OCaml.
- Have a knowledge about generating LR(1) parsers from BNF grammars or Parser Combinator notations

### Compiler - Middle End

- Experienced in various kinds of (static) program analyses and transformations , e.g., implementing custom binary operators with customizable associativities and precedences, partial evaluations, forward reference resolutions, lexical/dynamic scoping analysis, syntactic macros, etc.
- Familiar with type inferences based on HM unification ,and capable of extending it with with row polymorphisms, instance resolutions, GADTs, etc.

### Compiler - Back End

- Familiar with syntaxes, semantics and some intrinsics of LLVM IR
- Familiar with CPython bytecode instructions and code objects, etc.
- Have some experience about code generation that targets LLVM IR or CPython bytecode

### Compiler - Others

- Experienced in making DSLs
- Familiar with low level data layouts
- Familiar with implementing high level language constructs (Module, Pattern Matching, Switch, Closure, etc.) for both compiled languages and interpreted languages.

### Functional Programming

- Familiar with type classes and higher kinded types , and have created several implementaions
- Understand and can make good use of CPS, Y-Combinators, simple untyped/typed lambda calculus
- Understand Monad related stuffs from Monoid to MonadTrans , and have a preference of monadic coding style

### Machine Learning

- Used to be familiar with commonly-used DL frameworks like PyTorch and Tensorflow, and capable of picking up again in a few minutes.
- Experienced in traditional ML toolchains like NumPy, Scikit-Learn, Pandas, Matplotlib, etc.
- Understand forward propogation and back propogation, capable of creating simple neural network frameworks
- Understand many traditional ML algorithms like KNN, K-Means, Decision Tree, Random Forest, Stacking, etc.
- 2016 CCF/DataFountain Agricultural Product Price Prediction rank 7/500+
- Have some knowledge about bioinformatics and NLP , familiar with feature extraction methods(PSSM, N-Gram, TF-IDF, etc.)
- Capable of taking advantage of ML in daily life , such as playing FGO and creating smart CLI(like lightweighted auto-jump)

## 📄 MISCELLANEOUS

---

- Blog: <https://thautwarm.github.io/Site-32/>
- PyPI: <https://pypi.org/user/thautwarm/>
- PyConChina 2018: [http://cn.pycon.org/2018/city\\_beijing.html](http://cn.pycon.org/2018/city_beijing.html), as a lecturer(called *NightyNight*)
- Open source contributions: contributed to some organizations such as *Microsoft*, *Python*  
Fetch the meaningful records from <https://thautwarm.github.io/Site-32/Others/contributions.html>
- Codewars: <https://www.codewars.com/users/NightyNight/>, basically merely prefer to finish the most difficult trainings