

NoEsis:

Privacy, modularity and knowledge transfer in LLM Adaptation

Rob Romijnders (romijndersrob@gmail.com)

+ Stefanos Laskaridis, Ali Shahin Shamsabadi, Hamed Haddadi

Internship @ Brave, privacy-first browser
Appeared at ICLR 2025 MCDC workshop



Current AI

More data (bitter lesson) but soon exhaust all internet [1]

Business have lots of internal data, not on the public internet

Businesses consist of departments

This work:

- Novel **Membership Inference Attack** on Mixture Models
- Modular training with differential privacy

Larger neural models, but maintain privacy

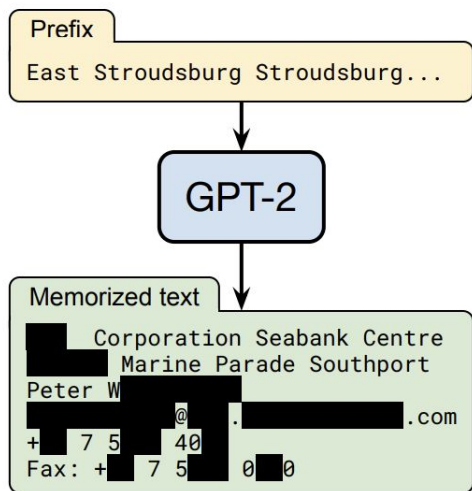
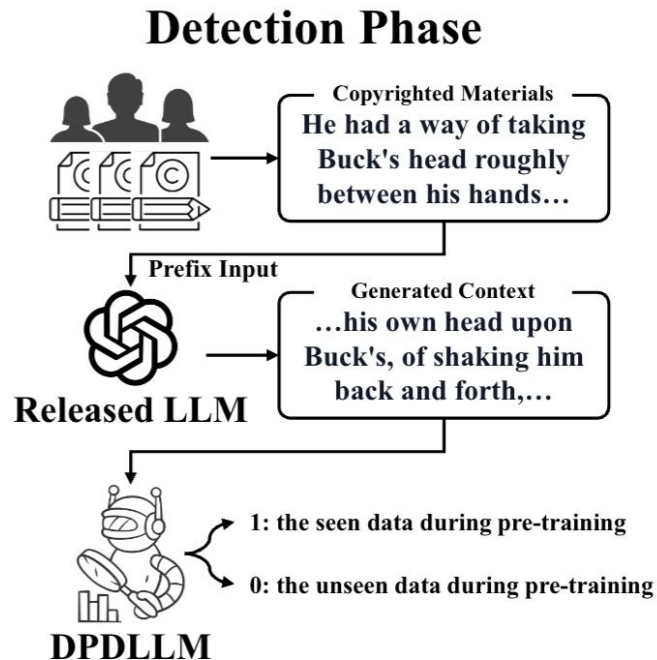


Figure 1: **Our extraction attack.** Given query access to a neural network language model, we extract an individual person's name, email address, phone number, fax number, and physical address. The example in this figure shows information that is all accurate so we redact it to protect privacy.



Why are multiple domains important?

- **LLM:**

multiple inhouse and external text datasets such as emails, law reviews

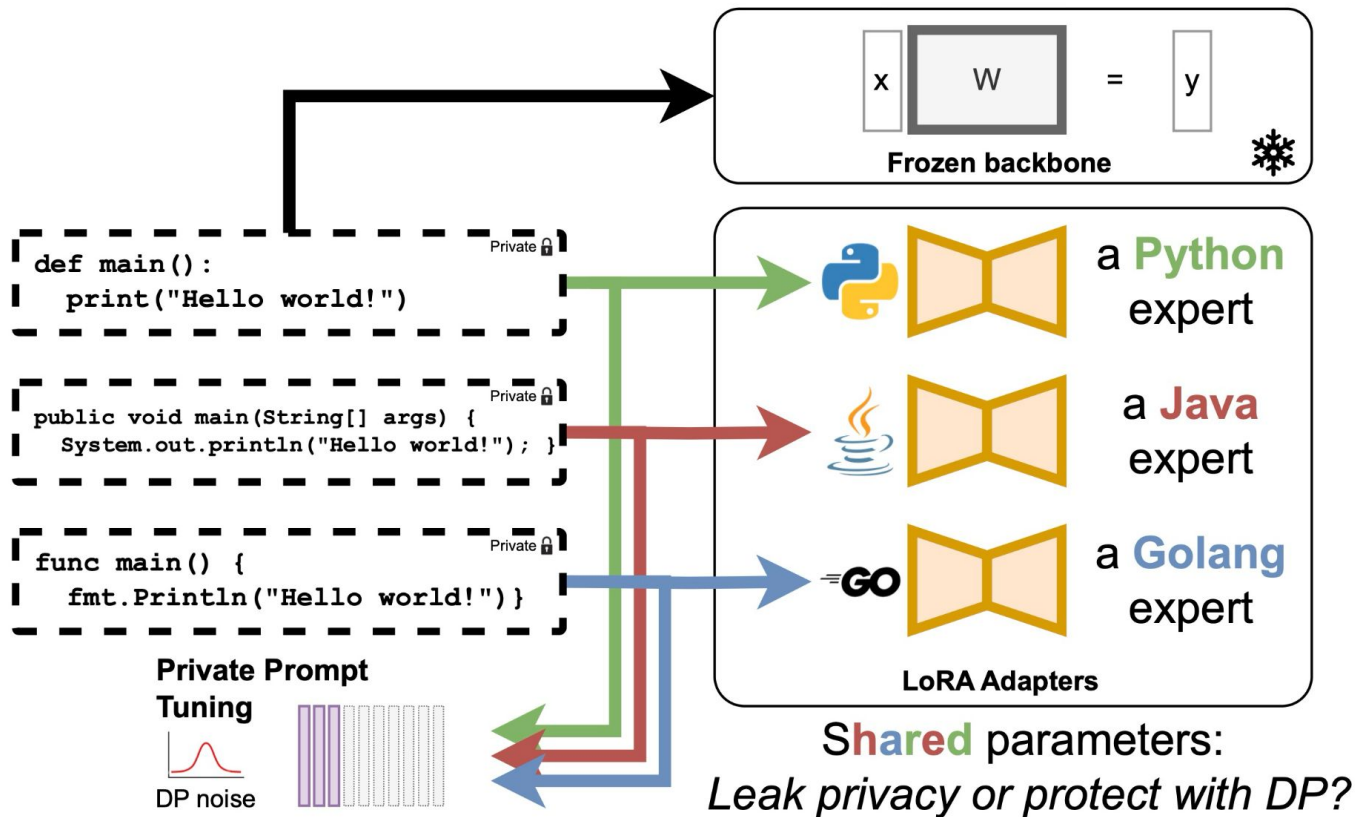
- **CodeLLM:**

external codebases on github, but combine with inhouse code

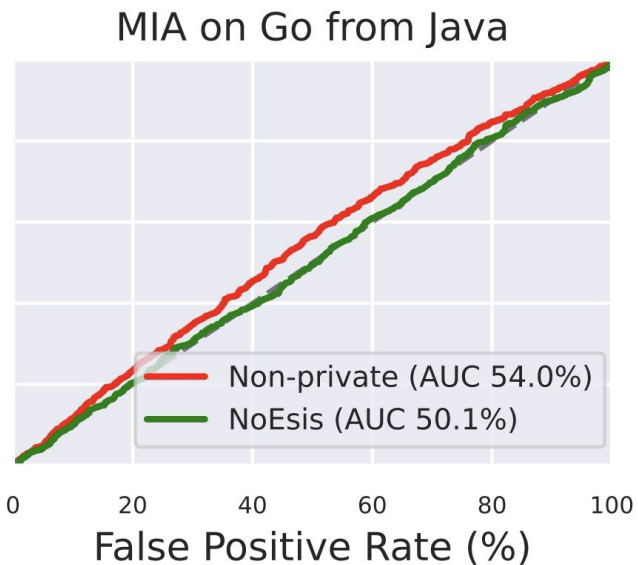
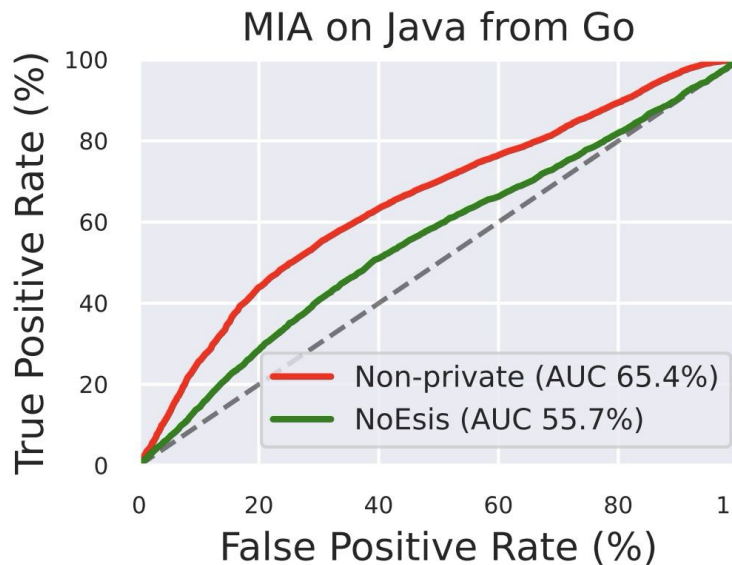
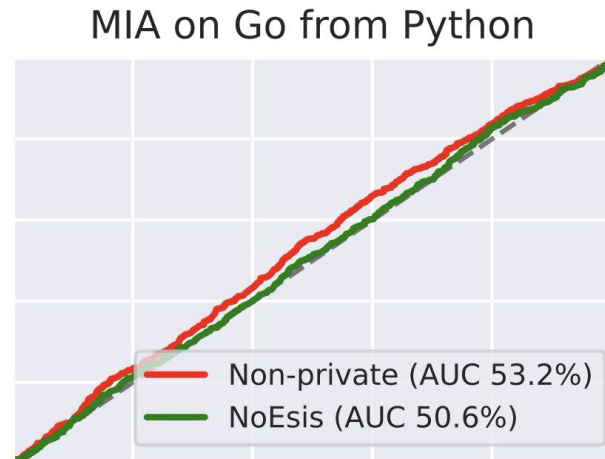
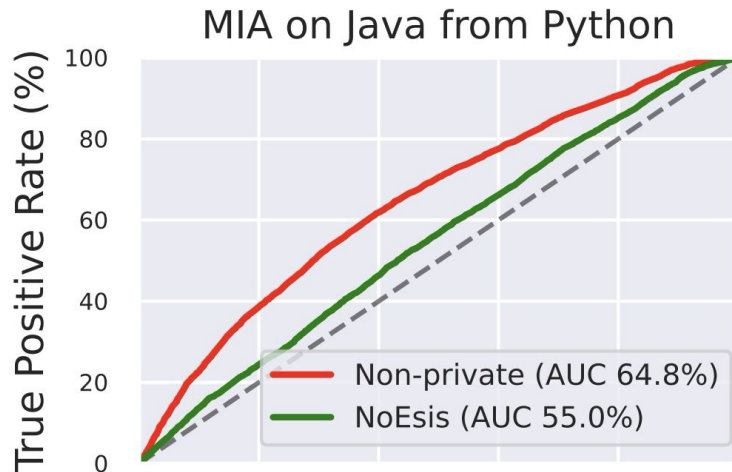
- **VLM:**

vision language models on paired image-text dataset, but restricted copyright in different countries

DP on the common parameters



MIA



Comparison of results

Knowledge transfer between departments

Privacy: learn patterns but don't learn details

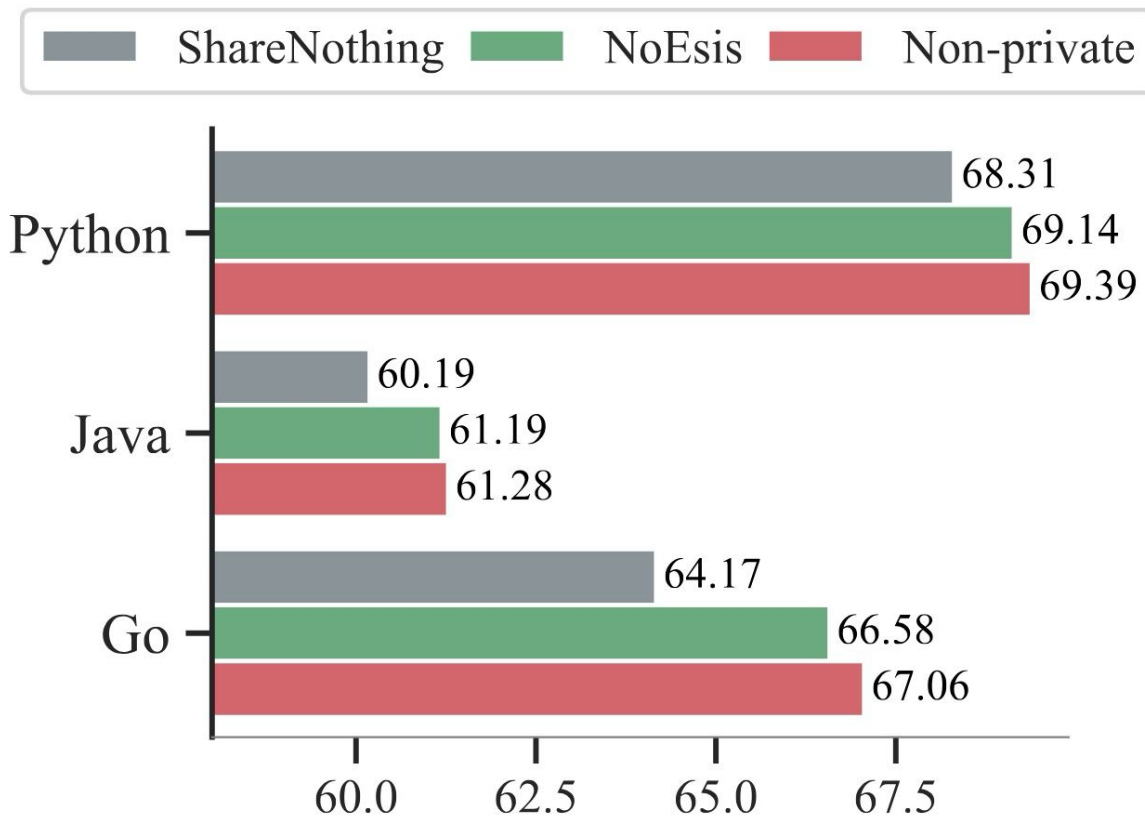
Modularity: easy to switch in and out, bandwidth, keep base model intact

Model	ϵ	Modular	Private	Transfer	PEFT	Python	Java	Go
(i) Share Nothing	0.0	✓	✓	✗	✓	68.31	60.19	64.17
(ii) Solo (separate models)	1.0	✗	✓	✗	✗	30.19	14.84	4.84
(iii) Monolithic Fine-tuning (Abadi et al., 2016)	1.0	✗	✓	✓	✗	36.05	23.54	18.34
(iv) Single Common Adapter (rc=512)* (Yu et al., 2021b)	1.0	✗	✓	✓	✓	48.21	36.16	27.24
(v) Prompt-Tuning Only (pt32) [†] (Duan et al., 2023)	1.0	✗	✓	✓	✓	35.03	24.29	9.67
(vi) NoESIS (pt32) [†]	1.0	✓	✓	✓	✓	69.14	61.18	66.53

*rc, rank r of the common LoRA; [†]pt: number of trainable prompt tokens

Bridge the gap

Bridge the gap between non-sharing and non-private



Conclusion

An architecture to train LLMs in multi-department institutions

- Knowledge transfer
- Differential Privacy
- Modularity

First work to show Membership Inference Attack for shared parameters in Mixture-of-Expert models

romijndersrob@gmail.com

github.com/RobRomijnders/noesis

Why modularity?

DP reasons:

- Generally, backprop on less weights suffers less from noise

Computational reasons:

- No large model updates to ship around

Hosting reasons:

- Hosting can optimize for model as-is

Java[tokenized]: example

```
[<s>, 'package', 'org', '', 'sqlproc', '', 'dsl', '', 'ui', '', 'import', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'DefaultUiModule', '', 'import', 'org', '', 'eclipse', '', 'ui', '', 'plugin', '', 'AbstractUIPlugin', '', '@', 'SuppressWarnings', '(', '"all"', ')', 'public', 'abstract', 'class', 'AbstractProcessorDslUiModule', 'extends', 'DefaultUiModule', '{', 'public', 'AbstractProcessorDslUiModule', '(', 'AbstractUIPlugin', 'plugin', ')', '{', 'super', '(', 'plugin', ')', '', '}', 'public', 'com', '', 'google', '', 'inject', '', 'Provider', '<', 'org', '', 'eclipse', '', 'xtext', '', 'resource', '', 'containers', '', 'IAllContainersState', '>', 'provideIAllContainersState', '(', ')', '{', 'return', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'shared', '', 'Access', '', 'getJavaProjectsState', '(', ')', '', '}', 'public', 'Class', '<', '?', 'extends', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'editor', '', 'contentassist', '', 'IProposalConflictHelper', '>', 'bindIProposalConflictHelper', '(', ')', '{', 'return', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'editor', '', 'contentassist', '', 'antlr', '', 'AntlrProposalConflictHelper', '', 'class', '', '}', 'public', 'void', 'configureHighlightingLexer', '(', 'com', '', 'google', '', 'inject', '', 'Binder', 'binder', ')', '{', 'binder', '', 'bind', '(', 'org', '', 'eclipse', '', 'xtext', '', 'parser', '', 'antlr', '', 'Lexer', '', 'class', ')', '', 'annotatedWith', '(', 'com', '', 'google', '', 'inject', '', 'name', '', 'Names', '', 'named', '(', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'LexerUIBindings', '', 'HIGHLIGHTING', ')', ')', '', 'to', '(', 'org', '', 'sqlproc', '', 'dsl', '', 'parser', '', 'antlr', '', 'internal', '', 'InternalProcessorDslLexer', '', 'class', ')', '', '}', 'public', 'void', 'configureHighlightingTokenDefProvider', '(', 'com', '', 'google', '', 'inject', '', 'Binder', 'binder', ')', '{', 'binder', '', 'bind', '(', 'org', '', 'eclipse', '', 'xtext', '', 'parser', '', 'antlr', '', 'ITokenDefProvider', '', 'class', ')', '', 'annotatedWith', '(', 'com', '', 'google', '', 'inject', '', 'name', '', 'Names', '', 'named', '(', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'LexerUIBindings', '', 'HIGHLIGHTING', ')', ')', '', 'to', '(', 'org', '', 'eclipse', '', 'xtext', '', 'parser', '', 'antlr', '', 'AntlrTokenDefProvider', '', 'class', ')', '', '}', 'public', 'Class', '<', '?', 'extends', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'refactoring', '', 'IDependentElementsCalculator', '>', 'bindIDependentElementsCalculator', '(', ')', '{', 'return', 'org', '', 'eclipse', '', 'xtext', '', 'ui', '', 'refactoring', '', 'impl', '', 'DefaultDependentElementsCalculator', '', 'class', '', '}', 'public', 'void', 'configureResourceDescriptionsBuilderScope', '(', 'com', '', 'google', '', 'inject', '', 'Binder', 'binder', ')', '{', 'binder', '', 'bind', '(', 'org', '', 'eclipse', '', 'xtext', '', 'resource', '', 'IResourceDescriptions', '', 'class', ')', '', 'annotatedWith', '(', 'com', '', 'google', '', 'inject', '', 'name', '', 'Names', '', 'named', '(', 'org', '', 'eclipse', '', 'xtext', '',
```

Python[tokenized]: example 1

```
[<s>, 'import', 'os', <EOL>, 'import', 'sys', <EOL>, 'if', '__name__' == '__main__', ':', <EOL>, 'os', ':', 'environ', ':', 'setdefault', '(',  
'', ':', '', ')', <EOL>, 'from', 'django', ':', 'core', ':', 'management', 'import', 'execute_from_command_line', <EOL>,  
'execute_from_command_line', '(', 'sys', ':', 'argv', ')', </s>]
```

```
[<s>, 'from', '__future__', 'import', 'unicode_literals', <EOL>, 'from', 'django', ':', 'db', 'import', 'models', ':', 'migrations', <EOL>, 'class',  
'Migration', '(', 'migrations', ':', 'Migration', ')', ':', <EOL>, 'dependencies', '=', '[', <EOL>, ']', <EOL>, 'operations', '=', '[', <EOL>,  
'migrations', ':', 'CreateModel', '(', <EOL>, 'name', '=', '"Category"', ':', <EOL>, 'fields', '=', '[', <EOL>, '(', '"id"', ':', 'models', ':',  
'AutoField', '(', 'verbose_name', '=', '"ID"', ':', 'serialize', '=', 'False', ':', 'auto_created', '=', 'True', ':', 'primary_key', '=', 'True', ')', ':',  
<EOL>, '(', '"name"', ':', 'models', ':', 'CharField', '(', 'help_text', '=', '"b"', ':', 'max_length', '=', '40', ')', ':', <EOL>, '(', '"image"', ':',  
'models', ':', 'ImageField', '(', 'help_text', '=', '"b"', ':', 'null', '=', 'True', ':', 'upload_to', '=', '"b'categories"', ':', 'blank', '=', 'True', ')', ':',  
<EOL>, ']', ':', <EOL>, 'options', '=', '{', <EOL>, '"ordering"', ':', '(', '"name"', ':', ')', ':', <EOL>, '', ':', '"Categories"', ':', <EOL>, '}', ':',  
<EOL>, 'bases', '=', '(', 'models', ':', 'Model', ':', ')', ':', <EOL>, ')', ':', <EOL>, ']', </s>]
```

Python[tokenized]: example 2

```
[<s>, 'import', 'twitter', '<EOL>', 'from', 'django', '.', 'contrib', 'import', 'messages', '<EOL>', 'from', 'django', '.', 'contrib', '.', 'auth', '.',  
'decorators', 'import', 'user_passes_test', '<EOL>', 'from', 'django', '.', 'db', 'import', 'transaction', '<EOL>', 'from', 'django', '.', 'shortcuts',  
'import', 'redirect', ',', 'render', '<EOL>', 'from', 'twobuntu', '.', 'news', '.', 'forms', 'import', 'AddItemForm', '<EOL>', '@', 'user_passes_test',  
'(', 'lambda', 'u', ':', 'u', ':', 'is_staff', ')', '<EOL>', 'def', 'add', '(', 'request', ')', ':', '<EOL>', 'if', 'request', ':', 'method', '==', '"POST"', ':',  
<EOL>', 'form', '=', 'AddItemForm', '(', 'data', '=', 'request', ':', 'POST', ')', '<EOL>', 'if', 'form', ':', 'is_valid', '(', ')', ':', '<EOL>', 'item', '=',  
'form', ':', 'save', '(', 'commit', '=', 'False', ')', '<EOL>', 'item', ':', 'reporter', '=', 'request', ':', 'user', '<EOL>', 'try', ':', '<EOL>', 'with',  
'transaction', ':', 'atomic', '(', ')', ':', '<EOL>', 'item', ':', 'save', '(', ')', '<EOL>', 'except', 'twitter', ':', 'TwitterError', 'as', 'e', ':', '<EOL>',  
'messages', ':', 'error', '(', 'request', ':', '""', '%', 'e', ':', 'message', '[', '0', ']', '[', '"message"', ']', ')', '<EOL>', 'else', ':', '<EOL>', 'messages', ':',  
'info', '(', 'request', ':', '""', ')', '<EOL>', 'return', 'redirect', '(', '"home"', ')', '<EOL>', 'else', ':', '<EOL>', 'form', '=', 'AddItemForm', '(', ')',  
<EOL>', 'return', 'render', '(', 'request', ':', '"form.html"', ':', '{', '<EOL>', '"title"', ':', '"Add Item"', ':', '<EOL>', '"form"', ':', 'form', ':', '<EOL>',  
'"description"', ':', '""', ':', '<EOL>', '"action"', ':', '"Add"', ':', '<EOL>', '}', ')', </s>]
```

```
[<s>, '__all__', '=', '(', '""', ':', '""', ')', '<EOL>', 'class', 'DjangoWSGIException', '(', 'Exception', ')', ':', '<EOL>', 'pass', '<EOL>', 'class',  
'ApplicationCallError', '(', 'DjangoWSGIException', ')', ':', '<EOL>', 'pass', </s>]
```

Related work / gap in literature

“Configurable Foundation Models: Building LLMs from a Modular Perspective” ‘24

“Nexus: Specialization meets Adaptability for Efficiently Training Mixture of Experts” ‘24

- * **LLMs can be modular in FFN layers. Experts can be added at any point in time**

“MultiCoder: Multi-Programming-Lingual Pre-Training for Low-Resource Code Completion” ‘22

- * **Experts per programming language improves performance on code-completion tasks**

But what about privacy?

“Differentially Private Training of Mixture of Experts Models”, ‘24

- * **MoE can be trained under DP (token-level; workshop paper with lots of issues)**

Mixture of LoRA ubiquitous but leaks privacy


- * Wu et al. "[Mixture of LoRA Experts](#)." ICLR 2024
- * Feng et al. "[Mixture-of-LoRAs: An Efficient Multitask Tuning Method for Large Language Models](#)" LREC/COLING 2024
- * Li et al. "[MIXLORA: Enhancing Large Language Models Fine-Tuning with LoRA-based Mixture of Experts](#)" arxiv 2024
- * Shen et al., "[Multimodal Instruction Tuning with Conditional Mixture of LoRA](#)" ACL 2024

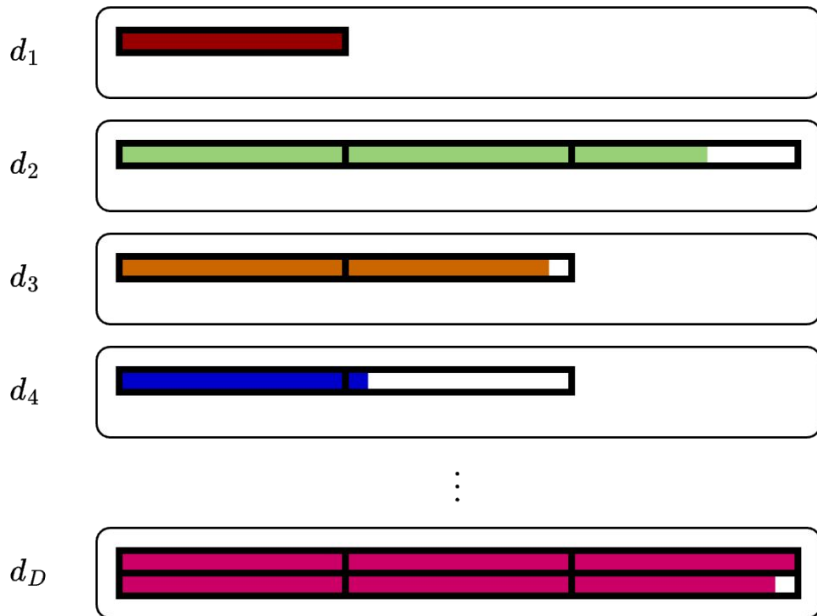
However, MixLoRA leaks privacy as evidenced by our *cross-domain MIA*.

When inserting DP as a defense, we find that DP on LoRA does not work well due to overparameterization and propose DP prompt-tuning as a low-parameter and privacy-preserving solution.

Document level DP

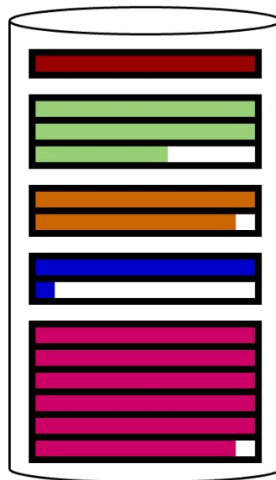
Each document, d_i , is a string of text
In total, D documents

 indicates a 'block', b_j of, c.f., 512 tokens, but the document can be of arbitrary size.



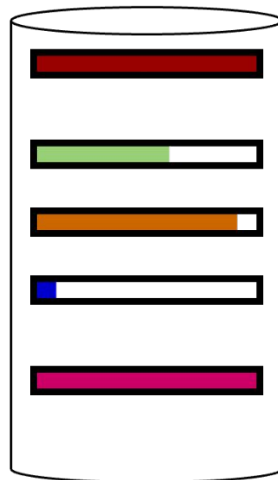
How to get dataset?

All blocks from all documents



- + Bigger dataset
- gradient accumulation
- epoch training takes longer

One blocks from each document



- Smaller dataset
- + Fast training
- Re-sample each epoch?

Python: example 2

```
def xmlstring(self, pretty_print=False):
```

```
    """Serialises this FoLiA element and all its contents to XML.
```

```
    Returns:
```

```
        str: a string with XML representation for this element and all its children"""
```

```
    s = ElementTree.tostring(self.xml(), xml_declaration=False, pretty_print=pretty_print, encoding='utf-8')
```

```
    if sys.version < '3':
```

```
        if isinstance(s, str):
```

```
            s = unicode(s,'utf-8') #pylint: disable=undefined-variable
```

```
    else:
```

```
        if isinstance(s,bytes):
```

```
            s = str(s,'utf-8')
```

```
    s = s.replace('ns0:',") #ugly patch to get rid of namespace prefix
```

```
    s = s.replace(':',ns0',"
```

```
    return s
```

Java: example 1

```
public static <T> T withStreams(Socket socket, @ClosureParams(value=SimpleType.class,  
options={"java.io.InputStream", "java.io.OutputStream"}) Closure<T> closure) throws IOException {  
    InputStream input = socket.getInputStream();  
    OutputStream output = socket.getOutputStream();  
    try {  
        T result = closure.call(new Object[]{input, output});  
        InputStream temp1 = input;  
        input = null;  
        temp1.close();  
        OutputStream temp2 = output;  
        output = null;  
        temp2.close();  
        return result;  
    } finally {  
        closeWithWarning(input);  
        closeWithWarning(output);  
    }  
}
```

Java: example 2

```
public void applyToPrimaryClassNodes(PrimaryClassNodeOperation body) throws CompilationFailedException {
    Iterator classNodes = getPrimaryClassNodes(body.needSortedInput()).iterator();
    while (classNodes.hasNext()) {
        SourceUnit context = null;
        try {
            ClassNode classNode = (ClassNode) classNodes.next();
            context = classNode.getModule().getContext();
            if (context == null || context.phase < phase || (context.phase == phase && !context.phaseComplete)) {
                int offset = 1;
                Iterator<InnerClassNode> iterator = classNode.getInnerClasses();
                while (iterator.hasNext()) {
                    iterator.next();
                    offset++;
                }
                body.call(context, new GeneratorContext(this.ast, offset), classNode);
            }
        } catch (CompilationFailedException e) {
            // fall through, getErrorReporter().failIfErrors() will trigger
        } catch (NullPointerException npe) {
            GroovyBugError gbe = new GroovyBugError("unexpected NullpointerException", npe);
            changeBugText(gbe, context);
            throw gbe;
        } catch (GroovyBugError e) {
```