Contents

Pre-Prompt Learning System **1**
- Overview . . . 1
- Key Concepts . . . 2
  - Attribution Analysis . . . 2
  - Learning Weights . . . 2
  - Exploration vs Exploitation . . . 2
- Admin Dashboard . . . 3
  - Overview Tab . . . 3
  - Templates Tab . . . 3
  - Attribution Tab . . . 3
  - Feedback Tab . . . 3
- Pre-Prompt Templates . . . 3
  - Default Templates . . . 3
  - Template Variables . . . 4
- Database Schema . . . 4
  - preprompt_templates . . . 4
  - preprompt_instances . . . 4
  - preprompt_feedback . . . 5
  - preprompt_attribution_scores . . . 5
- API Endpoints . . . 5
  - Dashboard . . . 5
  - Templates . . . 5
  - Feedback . . . 5
  - Learning Config . . . 5
- Integration with AGI Brain . . . 6
  - Code Example . . . 6
- Best Practices . . . 6
  - When to Adjust Weights . . . 6
  - Weight Adjustment Guidelines . . . 6
  - Monitoring Recommendations . . . 7
- Related Documentation . . . 7

Pre-Prompt Learning System

Version: 4.18.3
Last Updated: 2024-12-28

Overview

The Pre-Prompt Learning System tracks, evaluates, and learns from the effectiveness of pre-prompts (system prompts) used by the AGI Brain. Instead of blaming pre-prompts for all failures, it uses **attribution analysis** to understand what factor actually caused issues - whether it was the pre-prompt, model selection, orchestration mode, workflow, or domain detection.

Key Concepts

Attribution Analysis

When users provide feedback, the system doesn't just record whether the response was good or bad. It analyzes the full context to determine **what factor was most responsible**:

Factor	Description	When Blamed
Pre-prompt	System instructions were wrong	Tone, format, or approach mismatch
Model	AI model selection was inappropriate	Model lacks capability for task
Mode	Orchestration mode was wrong	Extended thinking when simple needed
Workflow	Workflow pattern didn't fit	Multi-step when single response needed
Domain	Domain detection was incorrect	Medical advice for cooking question
Other	External factors	User unclear, ambiguous request

Learning Weights

Each pre-prompt template has configurable weights that affect selection:

```
Final Score = Base + (Domain × DomainWeight) + (Mode × ModeWeight) +
              (Model × ModelWeight) + (Complexity × ComplexityWeight) +
              (TaskType × TaskTypeWeight) + FeedbackAdjustment
```

Weight	Default	Description
`baseEffectivenessScore`	0.5	Starting score
`domainWeight`	0.2	Bonus for matching domain
`modeWeight`	0.2	Bonus for matching mode
`modelWeight`	0.2	Bonus for compatible model
`complexityWeight`	0.15	Bonus for complexity match
`taskTypeWeight`	0.15	Bonus for task type match
`feedbackWeight`	0.1	Historical feedback influence

Exploration vs Exploitation

The system balances **exploitation** (using best-performing templates) with **exploration** (trying other templates to gather learning data):

- **Exploration Rate**: Percentage of requests where a non-optimal template is chosen
- **Default**: 10% exploration, decays over time
- **Minimum**: 1% to ensure continued learning

Admin Dashboard

Location: Admin Dashboard → Orchestration → Pre-Prompts
URL: `/orchestration/preprompts`

Overview Tab

- **Key Metrics**: Templates, Uses, Avg Rating, Thumbs Up Rate, Feedback Count
- **Attribution Pie Chart**: Visual breakdown of what gets blamed
- **Top Performing Templates**: Best-rated templates by feedback
- **Templates Needing Attention**: Low performers requiring adjustment

Templates Tab

- View all pre-prompt templates
- See usage statistics and success rates
- Adjust weights via slider interface
- View applicable modes and domains

Attribution Tab

- Detailed attribution breakdown
- Historical analysis of what factors contribute to success/failure
- Learning sample count

Feedback Tab

- Recent user feedback with ratings
- Attribution labels for each feedback
- Feedback text and timestamps

Pre-Prompt Templates

Default Templates

Template	Modes	Use Case
`standard_reasoning`	thinking, chain_of_thought	General questions
`extended_thinking`	extended_thinking	Complex reasoning
`coding_expert`	coding	Code generation
`creative_writing`	creative	Creative content
`research_synthesis`	research, analysis	Research tasks
`multi_model_consensus`	multi_model, self_consistency	Ensemble queries
`domain_expert`	all	Domain-specific expertise

Template Variables

Templates support `{{variable}}` placeholders:

Variable	Source	Example
`{{domain_name}}`	Domain detection	"Medicine"
`{{domain_confidence}}`	Detection confidence	"85"
`{{subspecialty_name}}`	Subspecialty	"Cardiology"
`{{field_name}}`	Field	"Healthcare"
`{{complexity}}`	Prompt analysis	"complex"
`{{task_type}}`	Task detection	"reasoning"
`{{key_topics}}`	Extracted topics	"heart, ECG, diagnosis"
`{{model_role}}`	For multi-model	"primary"
`{{proficiencies}}`	Domain proficiencies	"reasoning_depth: 9"

Database Schema

preprompt_templates

Stores reusable pre-prompt patterns.

Column	Type	Description
`template_code`	VARCHAR	Unique identifier
`system_prompt`	TEXT	Main prompt text
`context_template`	TEXT	Context with variables
`applicable_modes`	TEXT[]	Valid orchestration modes
`base_effectiveness_score`	DECIMAL	Base selection score
`*_weight`	DECIMAL	Selection weight factors
`total_uses`	INTEGER	Usage count
`avg_feedback_score`	DECIMAL	Average rating

preprompt_instances

Tracks actual pre-prompts used in plans.

Column	Type	Description
`plan_id`	UUID	Link to AGI plan
`template_id`	UUID	Template used
`full_preprompt`	TEXT	Rendered pre-prompt
`model_id`	VARCHAR	Model used
`orchestration_mode`	VARCHAR	Mode used
`detected_domain_id`	VARCHAR	Domain detected
`response_quality_score`	DECIMAL	Verification score

preprompt_feedback

User feedback with attribution.

Column	Type	Description
instance_id	UUID	Pre-prompt instance
rating	INTEGER	1-5 rating
thumbs_up	BOOLEAN	Simple feedback
issue_attribution	VARCHAR	What was blamed
issue_attribution_confidence	DECIMAL	Attribution confidence
feedback_text	TEXT	User comments

preprompt_attribution_scores

Learning data per template/factor combination.

Column	Type	Description
template_id	UUID	Template
factor_type	VARCHAR	model/mode/domain/etc
factor_value	VARCHAR	Specific value
success_correlation	DECIMAL	-1 to 1 correlation
sample_size	INTEGER	Data points
confidence	DECIMAL	Score confidence

API Endpoints

Dashboard

```
GET /api/admin/preprompts/dashboard
```

Returns dashboard data including metrics, attribution, top/low templates, recent feedback.

Templates

```
GET /api/admin/preprompts/templates
GET /api/admin/preprompts/templates/:id
PATCH /api/admin/preprompts/templates/:id/weights
```

Feedback

```
POST /api/admin/preprompts/feedback
GET /api/admin/preprompts/feedback/recent
```

Learning Config

```
GET /api/admin/preprompts/config
PATCH /api/admin/preprompts/config/:key
```

Integration with AGI Brain

The pre-prompt system integrates with `agi-brain-planner.service.ts`:

1. **Plan Generation**: Calls `prepromptLearningService.selectPreprompt()`
2. **Template Selection**: Scores templates based on context
3. **Variable Rendering**: Fills in `{{variables}}` from plan data
4. **Instance Tracking**: Records which template was used
5. **Feedback Loop**: User feedback updates attribution scores

Code Example

```
const prepromptResult = await prepromptLearningService.selectPreprompt({
  planId,
  tenantId,
  userId,
  orchestrationMode,
  modelId: primary.modelId,
  detectedDomainId: domainResult?.primary_domain?.domain_id,
  taskType: promptAnalysis.taskType,
  complexity: promptAnalysis.complexity,
  variables: {
    domain_name: domainResult?.primary_domain?.domain_name || 'general',
    complexity: promptAnalysis.complexity,
    // ... more variables
  },
});

plan.systemPrompt = prepromptResult.renderedPreprompt.full;
```

Best Practices

When to Adjust Weights

1. **Low feedback scores**: If a template consistently scores below 3.5
2. **High blame rate**: If pre-prompt is blamed >25% of the time
3. **Mode mismatch**: If template works well in some modes but not others

Weight Adjustment Guidelines

Situation	Adjustment
Template works better with specific models	Increase `modelWeight`
Template is mode-sensitive	Increase `modeWeight`
Domain expertise is critical	Increase `domainWeight`
Historical feedback is reliable	Increase `feedbackWeight`

Monitoring Recommendations

- Check attribution distribution weekly
- Review low-performing templates monthly
- Monitor exploration rate effectiveness
- Track thumbs-up rate trends

Related Documentation

- AGI Brain Plan System
- Orchestration Modes
- Domain Taxonomy
- Admin Guide - Orchestration