Appendix A

Optimization Model for 1-step Ahead Forecasting

Sets and indices:

$k = 1, \dots, K$:	index of the kth forecast source
$l = 1, \dots, L$:	index of the lth leading indicator
$s_k = 1, \dots, S_k$:	index of the s_kth constraint for the parameters of the forecast approach for kth forecast source
$t - \mathrm{D}, \dots, t$:	periods in the past
$t + 1$:	current period

Parameters:

DQ_{t-j}:	observed demand for $j = 0, \dots, D$
$FC_{k,t-j}$:	historical forecast for period $t - j, j = 0, \dots, D$, and forecast source $k = 1, \dots, K$
$I_{l,t-j}$:	historical leading indicator value for period $t - j, j = 0, \dots, D$, and leading indicator $l = 1, \dots, L$

Decision variables:

α_k:	weight for forecast value of the kth forecast source
β_l:	weight for the value of the lth leading indicator
γ_{kj}:	jth parameter to be chosen for the forecast approach of the kth forecast source.

The optimization formulation is stated as follows:

$$\min \sum_{i=0}^{D}\left\{\sum_{k=1}^{K} \alpha_k FC_{k,t-D+i}\left(\gamma_{k1}, \dots, \gamma_{kn_k}\right) + \sum_{l=1}^{L} \beta_l I_{l,t-D+i} - DQ_{t-D+i}\right\}^2 \quad (1)$$

subject to

$$\sum_{k=1}^{K} \alpha_k + \sum_{l=1}^{L} \beta_l \leq 1 \quad (2)$$

$$f_{ks_k}\left(\gamma_{k1}, \dots, \gamma_{kn_k}\right) = 0, k = 1, \dots, K, s_k = 1, \dots, S_k \quad (3)$$

$$\alpha_k \geq 0 \quad , \beta_l \geq 0, \gamma_{ki_k} \in D_{i_k}, k = 1, \dots, K, l = 1, \dots, L, i_k = 1, \dots, n_k. \quad (4)$$

The nonlinear objective function to be minimized, i.e. $F\left(\alpha_1, \dots, \alpha_K, \beta_1, \dots, \beta_L, \gamma_{11}, \dots, \gamma_{Kn_K}\right) \coloneqq \sum_{i=0}^{D}\left(\sum_{k=1}^{K} \alpha_k FC_{k,t-D+i}(\gamma_{k1}, \dots, \gamma_{kn_k}) + \sum_{l=1}^{L} \beta_l I_{l,t-D+i} - DQ_{t-D+i}\right)^2$, can be found in (1). It is the sum of the difference of the weighted combined forecast and leading indicator value for period $t - D + i$ and the corresponding observed demand quantity. (2) is a constraint for the weighting parameters for different forecast sources and leading indicators. Constraints for possible parameters of the different forecast approaches are modeled by constraint set (3). Here, $f_{ks_k}: \mathbb{R}^{n_k} \to \mathbb{R}$ are smooth functions. The non-negativity constraints for the decision variables α_k and β_l and the domains D_{i_k} for the decision variables γ_{ki_k} are expressed by (3).

Appendix B

LLM Usage

The LLM is used for two tasks. First, within FR4 the forecast DMA uses it for pattern matching, i.e. interpreting a product's input and output data to decide which forecasting approach fits best (e.g., exponential smoothing for stationary demand versus the Holt-Winters model for demand with trend and seasonality). Second, within the leading indicator candidate SA, it is used to align retrieved GDP values with the period length and aggregation level required by the current forecast (cf. Subsection 3.3). Note that the LLM does not perform clustering, the source and indicator clustering required by FR3 is carried out separately by the k-means algorithm (Pedregosa et al., 2011).

The GDP figures used as a leading indicator values are not recalled from the LLM's trained knowledge. The raw GDP figures are obtained through a dedicated MCP tool offered by the OECD organization. To avoid the look-ahead bias pointed out in review, the tool is required to query the provider's real-time (first-release) vintage rather than the latest revised figure. The publication lag itself is 30 days, which bounds how recent a GDP value can be.

The following prompt is used:

```
Your task is to retrieve the Gross Domestic Product (GDP) value needed as a
leading indicator for the current forecast, and rescale it to match the required
period length and aggregation level.

You have access to the following tool via MCP:
  query_gdp(country, period_start, period_end)

Note: query_gdp is connected exclusively to the OECD's real-time
(first-release) GDP series. Because GDP is published with a lag of approximately
30. Treat a missing result as an indication that the period is not yet published,
not as an error to retry.

Follow these steps:
1. Determine the geographic region and time period required for the
   current forecast, based on the request below.
2. Call query_gdp with the appropriate region and period range.
3. If the returned period length (monthly/quarterly) differs from the
   period length required by the current forecast, rescale the
   value(s) accordingly and state the method used (e.g., even
   division across months for quarterly-to-monthly, summation for
   monthly-to-quarterly).
4. If the required aggregation level is product-family rather than
   product level, apply the same rescaling factor uniformly since
   GDP has no product-level structure of its own.
5. Return only the rescaled value(s) in this format:
   {"gdp_value": <value>, "period": "<period>",
    "vintage": "first-release", "rescaling_applied": "<description>"}

Do not use any GDP figures from your own training data. Use only values returned
by the query_gdp tool.

Request: {leading_indicator_dma_request}
```

We compare Llama 3.1 70B hosted in Ollama v0.15.0 against GPT-5.5, accessed via the OpenAI API, on the two LLM-driven tasks described in the paper named the FR4 pattern-matching task and the GDP retrieval-and-rescaling task. For pattern matching, each model is

given the same input and output data for a product and is scored against a manually labeled ground truth (100%) demand pattern category (stationary, trend, …) for a sample of 45 products drawn from the data set described in Subsection 4.1. For the GDP task, both models are evaluated on a brief set of manually rescaled samples. Table 1 shows the belonging results of the GDP rescaling comparison.

Table 1: Accuracy of the LLM-based GDP retrieval and rescaling

Aggregation level	Sample size (periods)	Mean absolute percentage deviation
Daily, product-level	5	1.734
Weekly, product-level	5	0.845
Monthly, product-level	5	0.187

In addition to accuracy, we record per-call latency and monetary cost for the FR4 pattern matching task, since cost efficiency is central to the choice of an open-source model for a system that has to make many forecasting decisions at the same time (TR5).

The results show that the GPT 5.5 flagship model has higher accuracy and lower latency but at the same time much higher costs. In terms of accuracy, 92% is accurate enough for our experiments. For that reason, we chose to use the open-source variant instead of the commercial, paid solution from OpenAI.

Table 2: LLM comparison

Metric	Llama 3.1	GPT-5.5
FR4 pattern-matching accuracy	92%	94%
Mean latency per call	29.36 s	14.56 s
Cost per 1000 calls	$0	$77.64

Appendix C

Organization of the Agent Learning Over Time

FR10 requires the system to offer feedback/judgement mechanisms to learn appropriate forecast source and leading indicator combinations, and states that each agent receives reward information regarding its decisions. In addition, it is stated that we propagate each final forecasting result as feedback to all agents so that they can extract the necessary information and store it in their long-term memory.

The feedback stored for a completed forecast consists of the cluster, its underlying demand pattern, the forecast sources, the leading indicators, the aggregation level applied, and the resulting performance measure value. The reward information required by FR10 is derived directly from the IN_BOUNDS check in Algorithm 1. It is a combination that keeps the performance measure within the planners prescribed threshold. If it is inbound, it is treated as a positive outcome for the cluster it was applied to whereas a combination that requires a further iteration or leads to an infeasible result, is treated as a negative outcome.

This feedback directly drives the CLUSTER_SOURCES and CLUSTER_INDICATORS steps of Algorithm 1. Both take the accumulated history as an argument. Since the offline phase is repeated in a rolling horizon manner at the beginning of each planning epoch, the k-means algorithm (Pedregosa et al., 2011) is refit on a large history at every epoch which shifts the resulting cluster centroids (central point that represents a specific group) and membership. Each cluster carries a list of well promising forecast sources. This list is updated at each planning epoch by promoting forecast sources whose positive outcomes for the cluster have accumulated and demoting or removing forecast sources whose outcome for the cluster is negative. For instance, if the LSTM-based forecast source repeatedly leads to a performance measure value outside the prescribed bounds for demand patterns assigned to a given cluster, it is removed from that cluster's list of well promising sources in the following planning epoch. A forecast source that repeatedly stays within the bounds is retained. The same update rule applies to the leading indicator candidates.

Example: A family of similar products demand gets grouped into Cluster 7. Three candidate forecast sources are relevant, namely LSTM, exponential smoothing (ES), and human forecast, abbreviated by Human. The planner's threshold requires the performance measure (e.g. the Flores variant of the SMAPE) to be at least 75%. Table 3 is used to illustrate the example.

Table 3: Example for learning

Epoch	Cluster 7's source list entering this epoch	Sources tried	SMAPE	In bounds?	Stored in long-term memory
1	new cluster, no history yet.	{LSTM, ES, Human} "if no cluster is found, all forecast sources are used"	79%	Yes	Positive outcome for {LSTM, ES, Human} on Cluster 7. „Algorithm 1 returns at line 4"
2	{LSTM, ES, Human}	{LSTM, ES, Human}	61%	No	Negative outcome for {LSTM, ES, Human}. Algorithm 1 proceeds to line 5, adds the GDP indicator, and finds SMAPE = 77% (in bounds) but the optimization approach assigns LSTM's weight $\alpha \approx 0$ in that solution, i.e., LSTM contributed almost nothing
3	{LSTM, ES, Human}	Clusters list is reevaluated at the start of this epoch (k-means	84%	Yes	Positive outcome for {ES, Human},

	LSTM now has one success (epoch 1) and one near-zero-weight (epoch 2) association with a failing SMAPE (epoch 2)	refit on the now-larger history). LSTM is demoted. Sources used are {ES, Human}			LSTM is removed from Cluster 7's list
4	{ES, Human}	{ES, Human}	81%	Yes	Positive outcome again; list stays {ES, Human}

It is important to note that nothing is a trained policy. It is the k-means clustering being periodically refit on a growing history (so cluster membership of forecasting sources and leading indicators can change) and a simple check of in- and out-of-bounds per forecasting-source-cluster pair is performed.

Appendix D

Determination of Action Sequences

Algorithm 1 depicts the default sequence used by the combination DMA whenever no cluster-specific history is available for a given demand pattern. Each block of the algorithm adds to the combination used by the preceding block rather than replacing it. The default sequence always uses blocks first that leave the granularity of the forecasting target unchanged, i.e., adding forecast sources and/or leading indicators before blocks that change the granularity itself by aggregation.

Once a cluster has accumulated enough history, the combination DMA no longer falls back to the default sequence as mentioned before. Instead, the DMA starts learning and optimizing the sequence of steps by varying the sequence of them. Table 2 shows an example of how that learning is performed. The re-ranking process is not executed after a single epoch because otherwise a single noisy epoch could flip the order back and forth. In the example within the table, the threshold is set to two consecutive epochs that agree on which block was the one step that first produced an in-bounds result. In the table the block is named “product-agg” that leads two consecutive epochs to an in-bound SMAPE result. Therefore, in epoch 3 the procedure starts randomly alternating the steps by keeping the “product-agg” since it leads to valid results in the past epochs. Epoch 5 is doing the same since epoch 3 and 4 still confirm “product-agg” as beneficial.

Example: Consider a cluster, named Cluster 12, of product-family demand patterns whose month-to-month volatility is high but whose quarterly, family-level totals are stable. The planner's threshold requires the SMAPE performance measure to be at least 75% to be in bounds. The steps are shown in Table 4.

Table 4: Step-by-step learning example

Epoch	Block order used	Outcome per block (SMAPE, in bounds?)	Blocks needed	What gets updated?
1	*Default sequence*: sources → indicators → time-agg → product-agg (no Cluster 12 history yet)	sources 68% (no) → +indicators 71% (no) → +time-agg 74% (no) → +product-agg 83% (yes)	4	Cluster 12's history: product-aggregation was the block that first produced an in-bounds result
2	*Default sequence*: (still only one prior epoch, not yet enough to re-rank)	sources 66% (no) → +indicators 70% (no) → +time-agg 73% (no) → +product-agg 85% (yes)	4	Same pattern confirmed a second time for Cluster 12
3	*Re-ranked sequence*: sources → product-agg → indicators → time-agg (two consistent epochs now favor product-aggregation)	sources 69% (no) → +product-agg 85% (yes)	2	Cluster 12's block order is re-ranked. resolved in half as many blocks as epochs 1-2
4	*Re-ranked sequence stays:* (unchanged from Epoch 3)	sources 65% (no) → +product-agg 87% (yes)	2	Product-aggregation confirmed a third time as the decisive block; Cluster 12's block order remains unchanged
5	*Re-ranked sequence:* Sources → product-agg (two consistent epochs confirm product-aggregation as beneficial)	product-agg 70% (no)	1	Cluster 12's block order are re-ranked. It fails the threshold. It will revert.
6	*Reverted sequence:* (restored from Epoch 3)	sources 65% (no) → +product-agg 87% (yes)	2	Cluster is reverted to the previous sequence where the threshold is met.

This is one concrete instance of the pattern already observed in Subsection 4.2. Fewer computing activities are needed once the combination DMA has accumulated enough cluster-specific history, which is consistent with computational experiment scenario AITPA's lower computation time reported in Table 4 despite computational experiment scenario AITPA performing more types of computing activities per block than computational experiment scenario SFS.

Appendix E

Requirements Modeling

Table 5: Traceability linking requirements to their sources, implementation mechanism, and evaluation evidence

Requirement	Source	Implementation mechanism	Evaluation	Demonstration status (F/P/N)
FR1 (combine different forecasts and leading indicators)	Literature, industrial practice	Forecasting framework that allows to combine several forecast sources and leading indicator values, combination DMA	Computational experiments	F
FR2 (performance measurement)	Literature, industrial practice	Performance measurement SA	Computational experiments	F
FR3 (autonomous forecast source and leading indicator selection)	Industrial practice	Several DMAs and SAs	Computational experiments	P
FR4 (pattern detection)	Literature, industrial practice	Forecasting DMA	Computational experiments in a rolling horizon setting	N
FR5 (aggregation /disaggregation)	Literature, industrial practice	Combination DMA	Computational experiments	F
FR6 (information evolution)	Literature	Demand management SA	Computational experiments	N
FR7 (incorporation of human planners)	Literature, industrial practice	Forecasting framework that allows to combine several forecast sources and leading indicator values, combination DMA	Computational experiments	P
FR8 (integration of leading indicators)	Literature, industrial practice	Leading indicator DM and leading indicator agent candidate SA	Computational experiments	F

FR9 (goal fulfillment)	Own experience	Combination DMA	Computational experiments	P
FR10 (feedback/judgement)	Literature	MAS	Computational experiments	N
FR11 (what-if analysis)	Literature, industrial practice	Concept of DMAs and SAs	Computational experiments	N
TR1 (Scalability)	Literature, industrial practice	Cluster-based hardware-independent MAS deployment with LangChain and one EC2 instance per agent	Computational experiments	F
TR2 (HPC)	Literature, industrial practice	ML-based agents (LSTM training, k-means clustering) executed on an EC2 G4dn (multi GPU-based) instance	Computational experiments	F
TR3 (Communication abilities)	Literature	Google's A2A communication protocol, shared planning ontology, usage of Anthropic‘s MCP	Agent interactions demonstrated via UML diagrams	F
TR4 (automatic service invocation)	Literature, industrial practice	Model Context Protocol (MCP) tool integration	GDP retrieval-and-rescaling accuracy	F
TR5 (automated parameterization)	Literature, industrial practice	Forecasting approach parameterization SA. Llama 3.1 usage	Llama 3.1 vs. GPT-5.5 latency and cost comparison	F
TR6 (storage)	Literature	Short-term/long-term/permanent agent memory in LangGraph (part of Langchain framework)	Worked example demonstrating history-driven list updates across the process	F

Appendix F

Performance Assessment: Data for Computing the SMAPE Values and Organization of the Experiments

The real-world data set described in Subsection 4.1 (285 products, 48 months, from a large semiconductor company) is used entirely as the test set only and is never used during the offline training and parameterization phase.

The 750,000 synthetic instances are used exclusively for training and validation. Its split 80% for training and 20% for validation respectively.

To obtain statistically meaningful results, each scenario, i.e. SFS, AIS, AISTA, AITPA, and ALL is run for 20 independent replications. The performance measure value is recorded separately for each of the 20 replications, and confidence intervals are computed from these 20 values per scenario. Table 6 summarizes the configuration and the changes between the 20 independent replications. SMAPE lower-threshold is set to 75%. SMAPE upper-threshold is 100%.

Table 6: System configuration parameters

#	Component and parameters	Varies across the 20 replications? (Yes/No)	What exactly varies / how
1	Synthetic demand perturbation ~ U(-0.2, 0.2) used to build the 750,000 synthetic instances	Yes	Seed of randomizer set current date in ms.
2	Genetic algorithm used to solve the optimization problem popsize=10000, max_generations=1000, crossover_probability=0.1, mutation_probability=0.9, seed=random	Yes	Seed = random
4	k-means clustering used for CLUSTER_SOURCES and CLUSTER_INDICATORS in Algorithm 1. cluster_size=8, init=random, random_state=None Threshold= 2 consecutive epochs, SMAPE lower-threshold 75%	Yes	centroid initialization is randomized by default in scikit-learn
5	LSTM forecast source training Units=3, kernel_initializer= glorot_uniform, recurrent_initializer= orthogonal, seed=None, dropout=[0.0,0.0], global seed=random	Yes	weight initialization and batch shuffling are stochastic by default in Keras
6	80/20 train/validation split of the 750,000 synthetic instances	Yes	data can be different

7	Llama 3.1 sampling for FR4 pattern matching and GDP retrieval-and-rescaling Temperature=0.4, top_p=0.78, top_k=30, seed=0	no	

Appendix G

Additional Computational Experiments for a Single-pass Version of the Algorithms and a Version with Manually Chosen Weights

We add three non-agentic scenarios to the five agent-based forecasts already reported in the article. Single-pass is the model-centric forecasting framework as described in the article where the forecast is computed once with no iteration and no adaptive decision-making. Simple average follows Wang et al. (2023)'s finding that combination weights estimated from data are difficult to outperform with simple averaging. Manually chosen weights instead of fixed weights are used to mimic the situation where values are prescribed by a planner. In addition, we added the AFS3C-based scenario where we use our distributed agent-based approach. For the evaluation, we apply the same settings as described in the paper and in Appendix F. We use the following design of experiments from Table 7.

Table 7: Design of experiments

Factor	Level
Forecasting sources	LSTM, Exponential Smoothing, Human
Leading indicators	Order entry
Weights per scenario	**Single-pass**: - optimized **AFS3C-based**: - optimized **Simple average**: - 1/3 per source **Manually-weighted**: - LSTM=0.3 - Exponential Smoothing=0.2 - Human=0.5

Tables 7-8 show the results.

Table 8: SMAPE comparison including the non-agentic baseline

Scenario	SMAPE
Single-pass	68.98%
AITPA	83.13%
Simple average	46.45%
Manually-weighted	45.45%

Table 9: Execution time comparison including the non-agentic baseline

Scenario	Execution time
AITPA (central execution)	22.32 min
AITPA (distributed execution)	13.11 min

We see from Table 8 that optimizing the parameters leads to better SMAPE values than fixed or manually chosen parameters. In addition, we can see that the agent-based system (AITPA) performs best. In addition, the computing time in case of an agentic system is less than the one of the centralized version. It is reasonable since a distribution of the system leads to the same SMAPE values since it only adds more computing and memory capacity and does not change the algorithm logic. We only compare the execution time between the two AITPA versions since Table 8 already shows that it leads to the best SMAPE values.