NextGen Assisted Utility Specification: Multitask Reinforcement Learning for Transfer Across Datasets

Gabriel Nova*[1], Stephane Hess[2], and Sander van Cranenburgh[1]

[1]CityAI Lab, Transport and Logistics Group, Delft University of Technology, The Netherlands
[2]Institute for Transport Studies and Choice Modelling Centre, University of Leeds, UK

Short summary

Discrete choice model specification is a time-consuming task for modellers, who often specify and estimate multiple models while balancing fit, parsimony, and behavioural plausibility. We present Delphos, an Reinforcement learning agent that learns transferable specification strategies across transport choice datasets. Delphos frames specification as sequential decision-making problem in which it applies a sequence of modelling actions and receives rewards from an estimation environment based on goodness-of-fit and convergence. To transfer modelling decisions across datasets with different variable sets, we use a DeepSet-Q architecture that encodes specifications as sets of modelling terms and conditions decisions on a dataset context vector. This enables a shared policy to be trained in a multitask setting and to generalise its recommendations to unseen but related datasets. Delphos is trained on three mode choice datasets and increasingly proposes specifications that outperform the baselines over training. When then applied to an unseen dataset, the trained agent specifies competitive and behaviourally plausible models in less than 10 minutes on a CPU.

Keywords: Assisted choice model specification; Deep reinforcement learning; Artificial intelligence

1 Introduction

Specifying discrete choice models is an iterative, cognitively demanding, and time consuming process for choice modellers. In practice, modellers gradually construct utility specifications, estimate candidate models, and evaluate them by balancing goodness-of-fit, parsimony, and behavioural plausibility (Van Cranenburgh et al., 2022; Nova, van Cranenburgh, & Hess, 2025). This largely trial-and-error workflow has motivated the development of assisted specification approaches, which seek to support or partially automate the specification task. Two broad classes of approaches can be distinguished. The first comprises meta-heuristic methods that formulate specification as a combinatorial search problem and explore the space of possible models using optimisation techniques (Páez & Boisjoly, 2022; Rodrigues et al., 2020; Ortelli et al., 2021; Beeramoole et al., 2023; Haj-Yahia et al., n.d.). The second class involves machine-learning-based systems that aim to generate plausible model specifications by leveraging learnt modelling knowledge (Sfeir et al., 2025; Nova, Hess, & van Cranenburgh, 2025). Notably, Nova, Hess, & van Cranenburgh (2025) propose a reinforcement learning framework in which an agent is trained to iteratively construct discrete choice model specifications.

However, existing assisted specification approaches struggle to reuse modelling experience or knowledge across datasets. Meta-heuristic methods typically discard information about previously explored specifications upon termination of the search, offering little scope for cumulative learning. Current machine-learning-based approaches face a different limitation: they are generally trained and applied using representations tightly coupled to the variables of a specific dataset, which limits their transferability to new modelling contexts. As a result, each new dataset effectively requires the specification process to be learned from scratch.

To leverage transfer learning and improve the efficiency of assisted specification across choice modelling tasks, this paper builds on the reinforcement learning framework of Nova, Hess, & van Cranenburgh (2025) and extend Delphos to a multitask setting. We train Delphos across datasets to learn a shared decision rule that adapts to the modelling context by separating specification representation from decision-making. Specifically, Delphos encodes candidate models as sets of

modelling terms and combining this representation with the choice dataset context when selecting actions. This design allows the agent to reuse modelling experience across tasks by learning which modelling decisions tend to perform well under similar circumstances. The aim is to support the modeller by providing automated, data-driven suggestions for utility specifications, thereby reducing the cognitive and computational burden of manual trial-and-error search. In practical use, the agent functions as an intelligent assistant that interacts with the modeller's workflow, proposing high-quality, behaviourally plausible utility specifications in a targeted and efficient manner.

The remainder of the paper is organised as follows. Section 2 presents our reinforcement learning framework for assisted choice model specification across multiple datasets. Section 2.1 formulates the specification task as a Markov Decision Process, Section 2.2 describes the multitask extension that enables sharing modelling decisions across datasets, and Section 2.4 presents the training and evaluation protocol. Section 3 presents main results, and Section 4 provides some preliminary conclusions.

2 Methodological framework

This study introduces a reinforcement learning framework to automate the choice model specification process across multiple datasets. We first formulate the specification task as a Markov Decision Process, and then describe the multitask extension that enables sharing modelling decisions across datasets. Lastly, we describe the training and evaluation protocol used to assess learning and transferability.

2.1 Problem formulation

We formulate discrete choice model specification as a sequential decision problem within a Markov Decision Process (MDP) approach, in which an agent learns how to build utility specifications through interaction with an estimation environment (Figure 1). Within each episode, Delphos applies a sequence of modelling actions to update a specification and propose a final candidate. The episodes ends when the agent selects a terminate action, after which the environment estimates the candidate and returns modelling outcomes used to compute feedback based on goodness-of-fit, convergence, and behavioural expectations. Reinforcement learning addresses such problems by learning a policy that selects actions to maximise expected long-term reward (Bellman, 1957; Bertsekas, 2019; Littman, 1994).

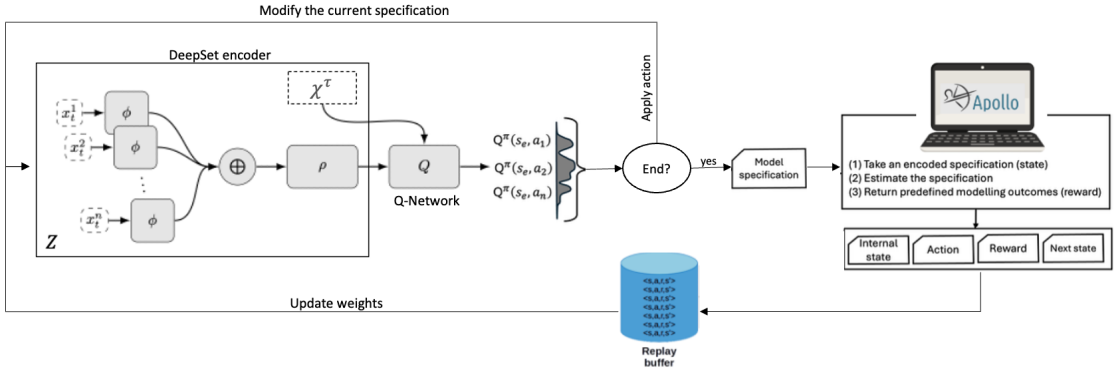

Figure 1: Delphos: A reinforcement learning framework for sharing modelling decisions across assisted choice model specification tasks. The agent takes a sequence of modelling actions to propose a specification, while the environment estimates it and returns a reward.

In our multitask setting, each task τ corresponds to specifying Multinomial Logit utility models (McFadden, 1978) for a choice dataset sampled from a given domain ($\tau \sim P(\tau)$). Each dataset thus induces its own MDP, defined by a task-specific state space ($\mathcal{S}_\tau$), action space ($\mathcal{A}_\tau$), transition function ($P(s_{\tau,e+1} \mid s_{\tau,e}, a_{\tau,e})$), and reward function ($\mathcal{R}_\tau$) (Burnetas & Katehakis, 1997; Sutton & Barto, 2018). We now describe the main components of multitask framework:

i) **State space**. It defines the set of all feasible model specifications that the agent may propose. It is built from a domain catalogue that determines the modelling components available to the agent across tasks:

1. Attributes: A set of attribute indices that may enter the utility specification, denoted as $[att_1, att_2, \ldots, att_K]$.
2. Transformations: A set of functional forms that can be applied to attributes to capture potential non-linearities, denoted by $[f_1, f_2, \ldots, f_T]$. In this study, we consider linear, logarithmic, and a Box–Cox transformations.
3. Taste parameters: An indicator of whether an attribute is associated with a generic or alternative-specific coefficient, denoted by [generic, specific]. A generic coefficient is shared across alternatives, while an alternative-specific coefficient varies across them.
4. Covariates: A set of variables that can interact with alternative-specific constants or attributes to capture observed heterogeneity, denoted by $[cov_1, cov_2, \ldots, cov_C]$.

Given this catalogue, a modelling term x_l specifies how a single attribute enters the utilities, and a model specification therefore corresponds to a unique state $s_e^\tau \in \mathcal{S}_\tau$, defined as the set of modelling terms accumulated through the agent's sequence of actions during episode e:

$$s_e^\tau = \{x_l\}_{l=1}^{L_e} = \{\,(k_1, t_1, g_1, c_1), \ldots, (k_{L_e}, t_{L_e}, g_{L_e}, c_{L_e})\,\} \tag{1}$$

where L_e denotes the number of modelling actions taken before termination. Each tuple (k_l, t_l, g_l, c_l) indexes the corresponding catalogue choices for attribute, transformation, taste structure, and covariate interaction, respectively. Unused components are defined by $t_l = g_l = c_l = 0$.

To illustrate this state representation, we now show how a utility specification can be expressed as a set of modelling terms using the tuple encoding in Eq. (1). Consider the utility for alternative i, where ASC_i is the alternative-specific constant, travel time (TT_i) enters with a logarithmic transformation and an alternative-specific coefficient, travel cost (TC_i) enters linearly with a generic coefficient that interacts with the covariate cov_1, and headway (HE_i) is not used:

$$V_i = ASC_i + \beta_{TT,i}\,\log(TT_i) + \Big(\beta_{C,\text{cov1\_0}}\text{cov}_1 == 0) + \beta_{C,\text{cov1\_1}}\text{cov}_1 == 1)\Big)\,TC_i$$

Using the encoding and defining the catalogue indices as $k \in \{0\text{: none}, 1\text{: } ASC, 2\text{: } TT, 3\text{: } TC, 4\text{: } HE\}$, $t \in \{1\text{: linear}, 2\text{: log}, 3\text{: Box–Cox}\}$, $g \in \{1\text{: generic}, 2\text{: alternative-specific}\}$, and $i \in \{0\text{: none}, 1\text{: cov}_1\}$, the resulting state is

$$s_e^\tau = \{(1,1,1,0),\ (2,2,2,0),\ (3,1,1,1),\ (4,0,0,0)\}.$$

ii) **Action space**. It defines all feasible operations that the agent can perform to update a specification. We consider three action types:

$$a \in \{\text{add}(k,t,g,c),\ \text{change}(k,t,g,c),\ \text{terminate}\}, \tag{2}$$

where *add* introduces a new modelling term, *change* modifies an existing term (e.g., transformation, taste type, or interaction), and *terminate* ends the episode (model specification). To avoid redundant or infeasible moves, such as selecting a covariate that is not available in task τ, actions are restricted to those that are valid given the current specification and dataset, using an action-masking mechanism (Huang & Ontañón, 2020).

iii) **Reward signal**. To support stable multitask training, the reward must be comparable and numerically stable across tasks. Raw goodness-of-fit values depend strongly on sample size, task complexity, and baseline fit (eg., null model or linear additive), which can distort the learning signal when a single policy is shared. We thus define the reward as the per-observation improvement in log-likelihood relative to a baseline model:

$$r = \tanh\Big(\frac{LL_{\hat{\beta}} - LL_{baseline}}{N_{obs}}\Big), \tag{3}$$

where the tanh transformation bounds the reward within $(-1, 1)$ to improve numerical stability, analogous to reward clipping (Mnih et al., 2015), and to maintain a comparable reward scale across

tasks in multitask settings (Hessel et al., 2019). We use the null model as the baseline, providing a meaningful reference and ensuring reward values remain comparable across heterogeneous datasets. Non-convergent models receive a fixed penalty (-1). Behavioural constraints can also be incorporated in the reward formulation as described by Nova, Hess, & van Cranenburgh (2025).

iv) **The environment**. It implements the estimation back-end. For a given dataset, it converts the agent's terminal state into an Apollo object (Hess & Palma, 2019), runs estimation, and returns the resulting modelling outcomes (e.g., log-likelihood and convergence properties) required to compute the reward.

2.2 Sharing modelling decisions across datasets

To enable transfer knowledge across datasets with different variable sets, we extend Delphos to a multitask setting and implement it as a DeepSet-Q network (Hügle et al., 2020) (Figure 2). In each training episode, Delphos is given a dataset τ with its corresponding vector x^τ that indicates which variables are available. It then takes a sequence of modelling actions to propose a model specification. Once the episode terminates, the environment estimates the candidate and returns a reward signal. By repeating this interaction across multiple datasets, Delphos learns a single shared policy that reuses modelling decisions that tend to perform well under similar dataset contexts.

Transfer is enabled by separating what has been specified from which dataset is being modelled. Specifically, the DeepSet-Q architecture (i) encodes model candidates in a variable-invariant embedding and (ii) conditions its actions on the dataset context. Formally, Delphos has two modules to separate specification encoding from context-dependent decision-making. First, a DeepSet encoder (Zaheer et al., 2017) maps the set of modelling terms $s_e^\tau = \{x_l\}_{l=1}^{L_e}$ to a fixed-length, permutation-invariant representation. Each modelling term x_l is embedded using a shared neural network $\phi(\cdot)$ and aggregated via pooling, $Z(s_e^\tau) = \rho(\sum_l \phi(x_l))$, which is invariant to the number and ordering of modelling terms. Second, this specification embedding is concatenated with the dataset context vector x^τ and fed to a Deep Q-network (Mnih et al., 2015) to compute action values, $q = Q(Z(s_e^\tau) \,||\, x^\tau)$. Invalid or redundant actions are masked before selection to ensure feasibility given the current specification and choice dataset.

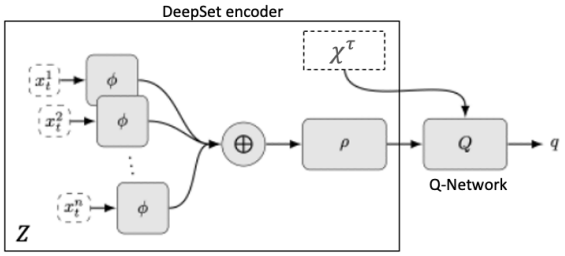

Figure 2: DeepSet-Q Network (adapted from Hügle et al. (2020))

This separation between the specification embedding and context-dependent decision-making enables a shared policy to generalise across datasets. As a result, Delphos can reuse modelling experience learned from related tasks when proposing specifications for unseen datasets. In the next subsection, we describe how the policy is trained across datasets.

2.3 Training and evaluation protocol

Delphos is trained in a balanced multi-task setting, where a single DeepSet-Q network is shared across all training datasets, but each action is conditioned on both the current specification and the choice dataset context. In each training episode, the agent takes a sequence of modelling actions to specify a candidate specification for each dataset. Upon estimation on their corresponding datasets, reward signals are computed using the Eq. (3). The resulting episode trajectories for each dataset are stored in a experience replay buffer along with their task ID, which are used for learning process (Mnih et al., 2015).

Policy updates are performed following the DeepSet Q-learning approach (Hügle et al., 2020) but ensuring each task contributes similarly to the loss function (Sutton et al., 1999; Sodhani et al., 2021). Specifically, we sample balanced mini-batches from the replay buffer to prevent dominance by datasets that use longer action trajectories simply because they contain more available variables and therefore require more internal specification steps. During training, ϵ-greedy exploration is gradually reduced to shift the agent from exploration towards the exploitation of actions associated with high-performing model specifications. Overall, this design aligns with the purpose of experience replay in deep reinforcement learning (Schaul et al., 2015) and with the idea of training should ensure balanced (Sodhani et al., 2021) and diverse exposure(Ross & Bagnell, 2010) across tasks.

To evaluate whether Delphos learns transferable modelling strategies, we compare the multi-task agent with agents trained from scratch when applied to unseen datasets. This allows us to test whether the dataset context helps guide the specification search in a new but related context. Then, we evaluate adaptation on unseen datasets while varying required episodes and which parts of the architecture are frozen (the specification embedding, the policy, or both). These analyses will show whether the agent can generalise beyond the training datasets and how easily it can be adapted to a new choice dataset with limited extra experience.

2.4 Experiments

We train Delphos on three transport mode choice datasets with share modelling principles but differing attributes and covariates: Apollo mode choice (Hess & Palma, 2019), Swiss route choice (Axhausen et al., 2008), DECISIONS (Calastri et al., 2020)). We then test transferability by using the learnt policy on an unseen dataset (Swissmetro, (Bierlaire et al., 2001)). We design our experiments to answer the folling questions: (i) Can Delphos transfer modelling strategies to unseen datasets? (ii) does dataset context improve specification search in new modelling context? (iii) how much additional training is needed for Delphos to adapt to an unseen dataset? (iv) Which components of the architecture matter most for speeding transfer? While our experiments so far cover (i) and (ii), we expect to show the full analysis at the conference and provide guidance for using Delphos on unseen choice datasets.

3 Results

3.1 Learning across training tasks

Delphos shows evidence of learning (Figure 3). Early in training, the agent often proposes specifications that do not outperform the baseline model, which reflects an exploration phase. As training progresses, Delphos starts specifying models that lie above the baseline one and, in later episodes, consistently outperform it. This pattern suggests that Delphos increasingly selects modelling actions that improve model fit by reusing experience gained across tasks.

3.2 Application to unseen data

When applied to the unseen Swissmetro dataset, Delphos proposes competitive candidate models within a budget of 100 model estimations in under 10 minutes on a laptop CPU. Within this budget, Delphos identifies the best specification with $LL_{best} = -0.71$ per observation. It not only outperforms the baseline ($LL_{linear} = -0.80$) but also is close to results reported by other assisted algorithms on the same dataset (e.g., $LL = -0.72$ per observation in Ortelli et al. (2021)).

We report the utility specification proposed by Delphos and estimated parameters in Table 1. It includes alternative-specific constants, considers relevant attributes with alternative-specific parameters, and introduces some non-linearities and meaningful covariate interactions. These results suggest that Delphos transfers modelling decisions to unseen datasets, reducing trial-and-error and reaching competitive, behaviourally plausible specifications in a small number of estimations. Importantly, inference runs on a standard laptop CPU, making it practical for analysts seeking to reduce specification overhead and for researchers interested in RL agents that learn reusable modelling strategies.

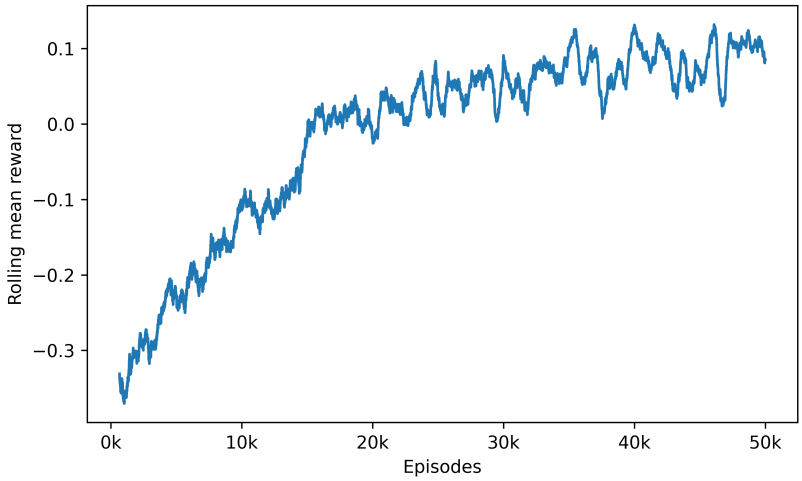

Figure 3: Rolling mean reward for Delphos during multitask training across choice datasets. Rewards are defined relative to the baseline; values above zero indicate better candidate models.

Table 1: Best-LL specification on Swissmetro

Parameter	**Attribute**	**Trans./Taste**	**Estimate (s.e.)**	***t*-rat.(0)**
$\mathrm{ASC}_{\mathrm{alt1}}$ (gender=0)			1.892 (0.142)	13.302
$\mathrm{ASC}_{\mathrm{alt1}}$ (gender=1)			0.802 (0.110)	7.273
$\mathrm{ASC}_{\mathrm{alt2}}$ (gender=0)			1.461 (0.152)	9.622
$\mathrm{ASC}_{\mathrm{alt2}}$ (gender=1)			1.099 (0.117)	9.386
λ_{TT}			0.625 (0.128)	4.888
$\beta_{\mathrm{alt1},TT}$ (age=1)	Travel time	Box–Cox / Specific	−0.674 (0.333)	−2.024
$\beta_{\mathrm{alt1},TT}$ (age=2)	Travel time	Box–Cox / Specific	−1.108 (0.195)	−5.681
$\beta_{\mathrm{alt1},TT}$ (age=3)	Travel time	Box–Cox / Specific	−1.756 (0.244)	−7.193
$\beta_{\mathrm{alt1},TT}$ (age=4)	Travel time	Box–Cox / Specific	−1.031 (0.243)	−4.240
$\beta_{\mathrm{alt1},TT}$ (age=5)	Travel time	Box–Cox / Specific	0.571 (0.287)	1.991
$\beta_{\mathrm{alt2},TT}$ (age=1)	Travel time	Box–Cox / Specific	−0.985 (0.419)	−2.348
$\beta_{\mathrm{alt2},TT}$ (age=2)	Travel time	Box–Cox / Specific	−1.705 (0.176)	−9.699
$\beta_{\mathrm{alt2},TT}$ (age=3)	Travel time	Box–Cox / Specific	−1.875 (0.148)	−12.680
$\beta_{\mathrm{alt2},TT}$ (age=4)	Travel time	Box–Cox / Specific	−0.312 (0.157)	−1.983
$\beta_{\mathrm{alt2},TT}$ (age=5)	Travel time	Box–Cox / Specific	0.142 (0.407)	0.349
$\beta_{\mathrm{alt3},TT}$ (age=1)	Travel time	Box–Cox / Specific	0.079 (0.123)	0.638
$\beta_{\mathrm{alt3},TT}$ (age=2)	Travel time	Box–Cox / Specific	−2.674 (0.205)	−13.070
$\beta_{\mathrm{alt3},TT}$ (age=3)	Travel time	Box–Cox / Specific	−2.417 (0.169)	−14.284
$\beta_{\mathrm{alt3},TT}$ (age=4)	Travel time	Box–Cox / Specific	−1.549 (0.188)	−8.221
$\beta_{\mathrm{alt3},TT}$ (age=5)	Travel time	Box–Cox / Specific	−1.192 (0.292)	−4.078
$\beta_{\mathrm{alt1},TC}$ (income=1)	Travel cost	Linear / Specific	−2.862 (0.208)	−13.730
$\beta_{\mathrm{alt1},TC}$ (income=2)	Travel cost	Linear / Specific	−2.804 (0.186)	−15.059
$\beta_{\mathrm{alt1},TC}$ (income=3)	Travel cost	Linear / Specific	−3.400 (0.200)	−17.010
$\beta_{\mathrm{alt1},TC}$ (income=4)	Travel cost	Linear / Specific	−2.000 (0.261)	−7.659
$\beta_{\mathrm{alt2},TC}$ (income=1)	Travel cost	Linear / Specific	−1.598 (0.117)	−13.628
$\beta_{\mathrm{alt2},TC}$ (income=2)	Travel cost	Linear / Specific	−1.285 (0.081)	−15.924
$\beta_{\mathrm{alt2},TC}$ (income=3)	Travel cost	Linear / Specific	−1.046 (0.063)	−16.567
$\beta_{\mathrm{alt2},TC}$ (inc=4)	Travel cost	Linear / Specific	−1.063 (0.165)	−6.434
β_{HE} (age=1)	Headway	Log / Generic	−1.902 (1.220)	−1.559
β_{HE} (age=2)	Headway	Log / Generic	−1.034 (0.591)	−1.749
β_{HE} (age=3)	Headway	Log / Generic	−1.016 (0.534)	−1.903
β_{HE} (age=4)	Headway	Log / Generic	0.056 (0.630)	0.088
β_{HE} (age=5)	Headway	Log / Generic	−2.309 (1.079)	−2.140
β_{SE} (gender=0)	Set availability	Log / Generic	−0.080 (0.210)	−0.380
β_{SE} (gender=1)	Set availability	Log / Generic	0.255 (0.203)	1.258
LL(0)				-5548.28
LL(final)				-3813.97

Parameter	Attribute	Trans. / Taste	Estimate (s.e.)	t-rat.(0)
AIC				7697.93
BIC				7928.79
Rho-squared				0.313
Adj. Rho-squared				0.306
N. observations				5409
Number of parameters				35

4 Conclusions

This paper extended Delphos to a multitask reinforcement learning setting to support assisted discrete choice model specification across related transport datasets. We formulate specification as a sequential decision problem in which a candidate specification is built through modelling actions and estimated using an estimation environment. Using a DeepSet-Q architecture, we represent each candidate specification as a set of modelling terms and combine it with dataset information, which enables to share the same decision rule across datasets even when they contain different variables. Our results show that Delphos learns reusable modelling strategies and increasingly outperforms the baseline specifications across the training tasks. When applied to an unseen dataset, Delphos transfers this experience and identifies competitive specifications within a limited number of estimations on standard CPU.

Despite these results, our multitask training setup has several limitations. First, we trained Delphos on three datasets and used for inference on a single unseen dataset, which limits conclusions about generalisation across similar modelling contexts. Second, we used an arbitrary reward clipping based on goodness-of-fit and convergence flag, which could be improved through explicit behavioural constraints. Future work will extend our experiment setup to a larger set of datasets, compare against task-specific agents trained from scratch, and evaluate few-shot fine-tuning on new datasets to determine when a shared policy improves learning speed and performance with few additional episodes.

Statistic	Value
LL(0)	-5548.28
LL(final)	-3813.97
AIC	7697.93
BIC	7928.79
Rho-squared	0.313
Adj. Rho-squared	0.306
N. observations	5409
Number of parameters	35
LL/N (Delphos)	-0.705
LL/N (VNS; Ortelli et al., 2021)	-0.720

Modelling term	Included	Taste	Transformation	Interaction
ASC	✓	Specific	–	gender
Travel Time	✓	Specific	Box–Cox	Age
Travel Cost	✓	Specific	Linear	Income
Headway	✓	Generic	Log	Age
Seat type	✓	Generic	Log	Gender

Modelling term	Included	Taste	Transformation	Interaction
ASC	✓	Specific	–	Gender
Travel time	✓	Specific	Box–Cox	Age
Travel cost	✓	Specific	Linear	Income
Headway	✓	Generic	Log	Age
Seat type	✓	Generic	Log	Gender
LL(0)				-5548.28
LL(final)				-3813.97
AIC				7697.93
BIC				7928.79
Rho-squared				0.313
Adj. Rho-squared				0.306
N. observations				5409
Number of parameters				35
LL/N (Delphos)				-0.710
LL/N (VNS; Ortelli et al., 2021)				-0.720

References

Axhausen, K. W., Hess, S., König, A., Abay, G., Bates, J. J., & Bierlaire, M. (2008). Income and distance elasticities of values of travel time savings: New swiss results. *Transport Policy*, *15*(3), 173–185.

Beeramoole, P. B., Arteaga, C., Pinz, A., Haque, M. M., & Paz, A. (2023). Extensive hypothesis testing for estimation of mixed-logit models. *Journal of choice modelling*, *47*, 100409.

Bellman, R. (1957). A markovian decision process. *Journal of mathematics and mechanics*, 679–684.

Bertsekas, D. (2019). *Reinforcement learning and optimal control* (Vol. 1). Athena Scientific.

Bierlaire, M., Axhausen, K., & Abay, G. (2001). The acceptance of modal innovation: The case of swissmetro. In *Swiss transport research conference.*

Burnetas, A. N., & Katehakis, M. N. (1997). Optimal adaptive policies for markov decision processes. *Mathematics of Operations Research*, *22*(1), 222–255.

Calastri, C., Crastes dit Sourd, R., & Hess, S. (2020). We want it all: experiences from a survey seeking to capture social network structures, lifetime events and short-term travel and activity planning. *Transportation*, *47*, 175–201.

Haj-Yahia, S., Mansour, O., & Toledo, T. (n.d.). Grammar-based approach to data-driven utility specification for discrete choice models. *Available at SSRN 5195530*.

Hess, S., & Palma, D. (2019). Apollo: A flexible, powerful and customisable freeware package for choice model estimation and application. *Journal of choice modelling*, *32*, 100170.

Hessel, M., Soyer, H., Espeholt, L., Czarnecki, W., Schmitt, S., & Van Hasselt, H. (2019). Multi-task deep reinforcement learning with popart. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 33, pp. 3796–3803).

Huang, S., & Ontañón, S. (2020). A closer look at invalid action masking in policy gradient algorithms. *arXiv preprint arXiv:2006.14171*.

Hügle, M., Kalweit, G., Werling, M., & Boedecker, J. (2020). Dynamic interaction-aware scene understanding for reinforcement learning in autonomous driving. In *2020 ieee international conference on robotics and automation (icra)* (pp. 4329–4335).

Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994* (pp. 157–163). Elsevier.

McFadden, D. (1978). Modelling the choice of residential location.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... others (2015). Human-level control through deep reinforcement learning. *nature*, *518*(7540), 529–533.

Nova, G., Hess, S., & van Cranenburgh, S. (2025). Delphos: A reinforcement learning framework for assisting discrete choice model specification. *arXiv preprint arXiv:2506.06410*.

Nova, G., van Cranenburgh, S., & Hess, S. (2025). Understanding the decision-making process of choice modellers. *Journal of choice modelling*, *56*, 100562.

Ortelli, N., Hillel, T., Pereira, F. C., de Lapparent, M., & Bierlaire, M. (2021). Assisted specification of discrete choice models. *Journal of choice modelling*, *39*, 100285.

Páez, A., & Boisjoly, G. (2022). *Discrete choice analysis with r.* Springer.

Rodrigues, F., Ortelli, N., Bierlaire, M., & Pereira, F. C. (2020). Bayesian automatic relevance determination for utility function specification in discrete choice models. *IEEE Transactions on Intelligent Transportation Systems*, *23*(4), 3126–3136.

Ross, S., & Bagnell, D. (2010). Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 661–668).

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

Sfeir, G., Nova, G., Hess, S., & van Cranenburgh, S. (2025). Can large language models assist choice modelling? insights into prompting strategies and current models capabilities. *arXiv preprint arXiv:2507.21790*.

Sodhani, S., Zhang, A., & Pineau, J. (2021). Multi-task reinforcement learning with context-based representations. In *International conference on machine learning* (pp. 9767–9779).

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. *A Bradford Book*.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, *112*(1-2), 181–211.

Van Cranenburgh, S., Wang, S., Vij, A., Pereira, F., & Walker, J. (2022). Choice modelling in the age of machine learning-discussion paper. *Journal of choice modelling*, *42*, 100340.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., & Smola, A. J. (2017). Deep sets. *Advances in neural information processing systems*, *30*.