Software Engineering and Applications 软件工程与应用, 2019, 8(6), 364-371
Published Online December 2019 in Hans. http://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2019.86045

Hans汉斯

Research on Computing Offloading Based on Data Stream Correlation in Industrial Intelligent Edge Computing

Ying Wang, Yifei Li

A School of Information, Zhejiang Sci-Tech University, Hangzhou Zhejiang
Email: 919164956@qq.com, yifeili1002@gmail.com

Received: Nov. 27th, 2019; accepted: Dec. 12th, 2019; published: Dec. 19th, 2019

Abstract

In order to meet the high real-time demand under the background of industrial Internet of Things in the era of big data, we propose a task offloading method based on data stream correlation in industrial intelligent edge computing based on edge computing and machine learning. Topological sorting, decision tree are adopted to seek task relevance, so as to avoid problems such as excessive cache and insufficient memory capacity caused by irrelevant data offloading, repeat data offloading multiple times and unordered offloading in multi-task scenarios. In this study, two scheduling algorithms are designed, respectively corresponding to two different scenarios of high and low repetition rate of characteristic data required by multi-task decision so as to ensure the real-time performance of task processing. In addition, this study is strictly controlled under the constraint of bandwidth to ensure the reliability of scheduling.

Keywords

Edge Computing, Machine Learning, Industrial Internet of Things, Computation Offloading

工业智能边缘计算中基于数据流相关性的计算卸载研究

王　莹，李逸飞

浙江理工大学信息学院，浙江　杭州
Email: 919164956@qq.com, yifeili1002@gmail.com

收稿日期：2019年11月27日；录用日期：2019年12月12日；发布日期：2019年12月19日

文章引用：王莹，李逸飞．工业智能边缘计算中基于数据流相关性的计算卸载研究[J]．软件工程与应用，2019，8(6): 364-371. DOI: 10.12677/sea.2019.86045

摘　要

大数据时代背景下为满足工业物联网的高实时性需求，我们运用边缘计算和机器学习技术提出了一种工业智能边缘计算中基于数据流相关性的计算卸载方法。采用拓扑排序、决策树等方式寻求任务相关性，避免无关数据卸载、多任务场景下重复数据多次卸载及无序卸载导致的缓存过多，存储器容量不足等问题。本研究设计了两种调度算法，分别对应多任务决策所需特征数据重复率高和低的两种不同场景下所采用的计算卸载算法，保证了任务处理的实时性；此外本研究均严格控制在带宽约束下，保证了调度的可靠性。

关键词

边缘计算，机器学习，工业物联网，计算卸载

1. 引言

随着大数据的不断发展，数字化的思想逐步渗透到工业制造业中。工业 4.0 趋势[1]就是将更多的物联网设备部署到工厂中，用算法取代人脑，用自动化设备取代人力，逐步实现工业智能化。然而，不可忽视的是随着联网设备的不断扩充，多用户多任务并行处理需求。传统集中式云计算已无法满足当下工业生产的实时性需求。边缘计算将部分计算卸载到边缘，在边缘端实现对数据流的简单计算或筛选[2] [3]，减轻云端计算及存储压力。同时由于计算位置上的变化，边缘设备更靠近用户端，可实现短时间内的有效交互，大大降低传输时延。边缘计算的出现为时间敏感型工业生产带来福音。

据 IDC 数据统计，到 2020 年将会有超过 500 亿的终端联网设备，超过 50%的数据需要在网络边缘侧分析、处理与储存。边缘服务器往往是由较近的路由器，基站等设备构成，虽然能为云端解压，但受其有限的计算及存储能力限制，无法满足大型任务需求。针对目前多用户多任务并行处理的现况，必须有一套合理的计算卸载方案来保证边缘服务器计算的可靠性。

现有方案中，Yaozhong Song [4]等人基于 QoS 面向物联网任务提出了一种在边缘计算网络中定期分配传入任务的方法，增加了可以在边缘计算网络中处理的任务数量，然而其并没有考虑到任务之间的关联性，及任务之间的优先级关系。Feng Wei [5]等人面向移动边缘计算物联网设备之间分配资源提出了一种选择最低能耗优先的算法，充分考虑服务器资源及无线信道分配，使用户端到边缘端的任务分配更加科学。然而其并没有讨论带宽对分配方案的影响，也没有讨论任务间的相关性。Zhang D [6]等针对异构边缘计算设计了一种基于社交传感的任务分配方法，考虑了任务的动态性，然而其也没有考虑任务的关联性并且仅能用于某一单独的任务，无法满足多任务并存场景。Boutheina Dab [7]等面向边缘计算的多任务联合分配及资源分配提出基于多用户 WiFi 的 MEC 架构，满足任务延迟约束下移动边缘端的能量消耗最小，他们默认服务器接收所有数据，但由于边缘服务器计算能力存储能力有限，这一想法虽然能短期内使时延及能耗最小化，但在大数据任务中将很难实现。

现有关于边缘计算的计算卸载问题大多考虑任务截止时间，容量，计算能力等限制，根据指定优先级进行任务分配。工厂往往部署多个传感器收集不同的特征数据，利用机器学习决策树思想发现，任务中常

存在很多决策无关的特征数据流，放弃无关数据的处理不仅可避免决策无关数据的传输开销，而且能减轻边缘端计算及存储压力，同时由于先到的处理优先级低的特征数据必须进行缓存等待优先级高的特征数据流到达，所以按特征数据流的优先级进行传输，可避免边缘服务器缓存过多导致容量不足的问题。

本文将基于边缘计算框架，联系机器学习算法和多数据流思想，对问题进行深入研究。

2. 系统模型

智能边缘计算的架构如图 1 所示，考虑多个任务集合，每个任务都含有若干个特征数据流。多传感器作为用户端，各自接收数据组成一个特征数据流。云端根据多任务需求生成决策树，并将训练好的模型放置边缘服务器上[8] [9]。各个边缘服务器依照自身计算能力和存储能力选择是否接收用户端的计算卸载请求。

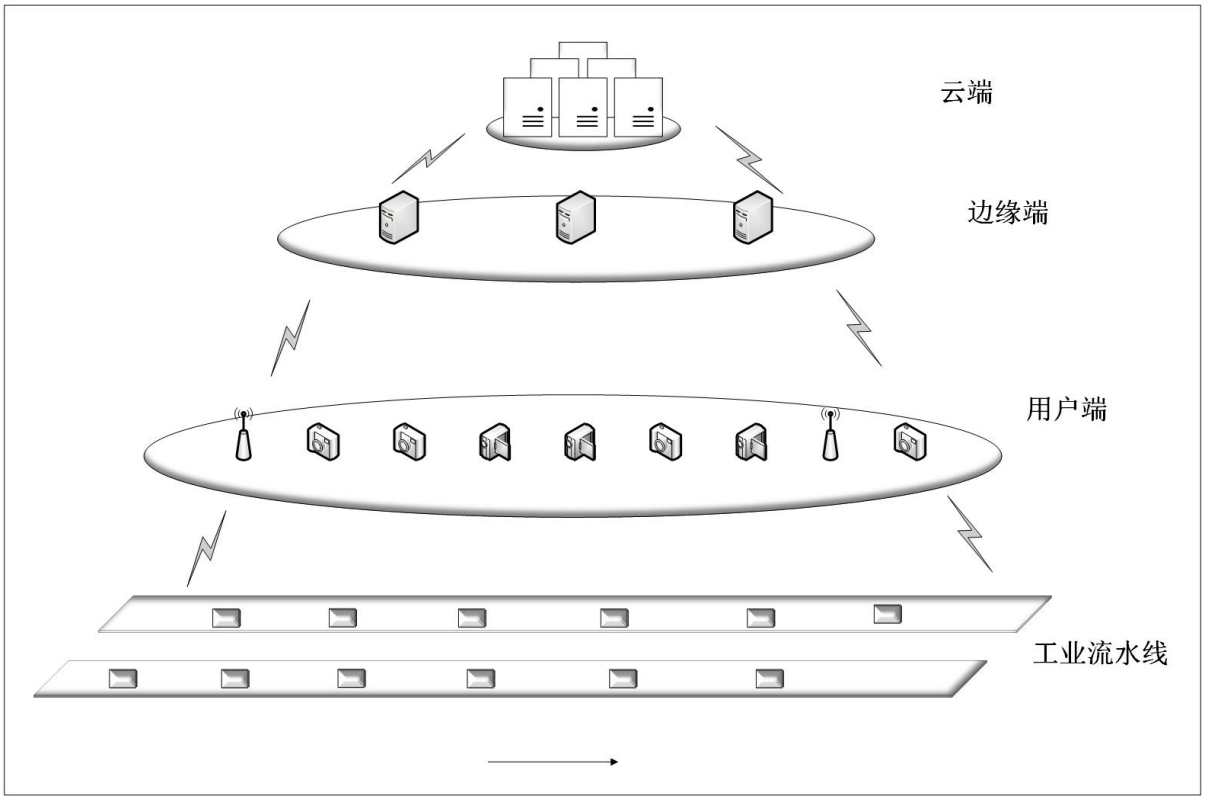

Figure 1. Industrial intelligent edge computing framework
图 1. 工业智能边缘计算框架

3. 问题描述

为满足当下工业生产实时性需要，考虑任务相关性，在 MEC (Multi-Access Edge Computing)多接入边缘服务器计算能力存储能力限制下，设计满足多个任务并存场景的计算卸载方案。应用场景图如图 2。

本文以多任务平均决策时间为优化目标[10] [11]，首先对基于数据流相关性面向工业智能边缘计算的任务卸载方法进行建模。考虑 $A=\{a_1,a_2,\cdots,a_n\}$ 为一个数量为 n 的任务集合，每个任务 a_w 都含有 q_n 个特征数据流，由 $F_m=\{f_1,f_2,\cdots,f_{q_n}\}$ 表示，每个 $f_i\in F_m$。B_i 表示特征 f_i 调度到边缘服务器 e 上所需带宽，d_i 表示特征数据流 f_i 的数据量，f_i 常用 $[B_i,d_i]$ 表示。在不同计算调度方案中，f_i 的卸载顺序不同，因此引入次序的概念，用字母 m 表示。t_s^m 为第 m 个特征数据流传输开始时间，t_e^m 为第 m 个特征数据流传输结束时间。由下式表示：

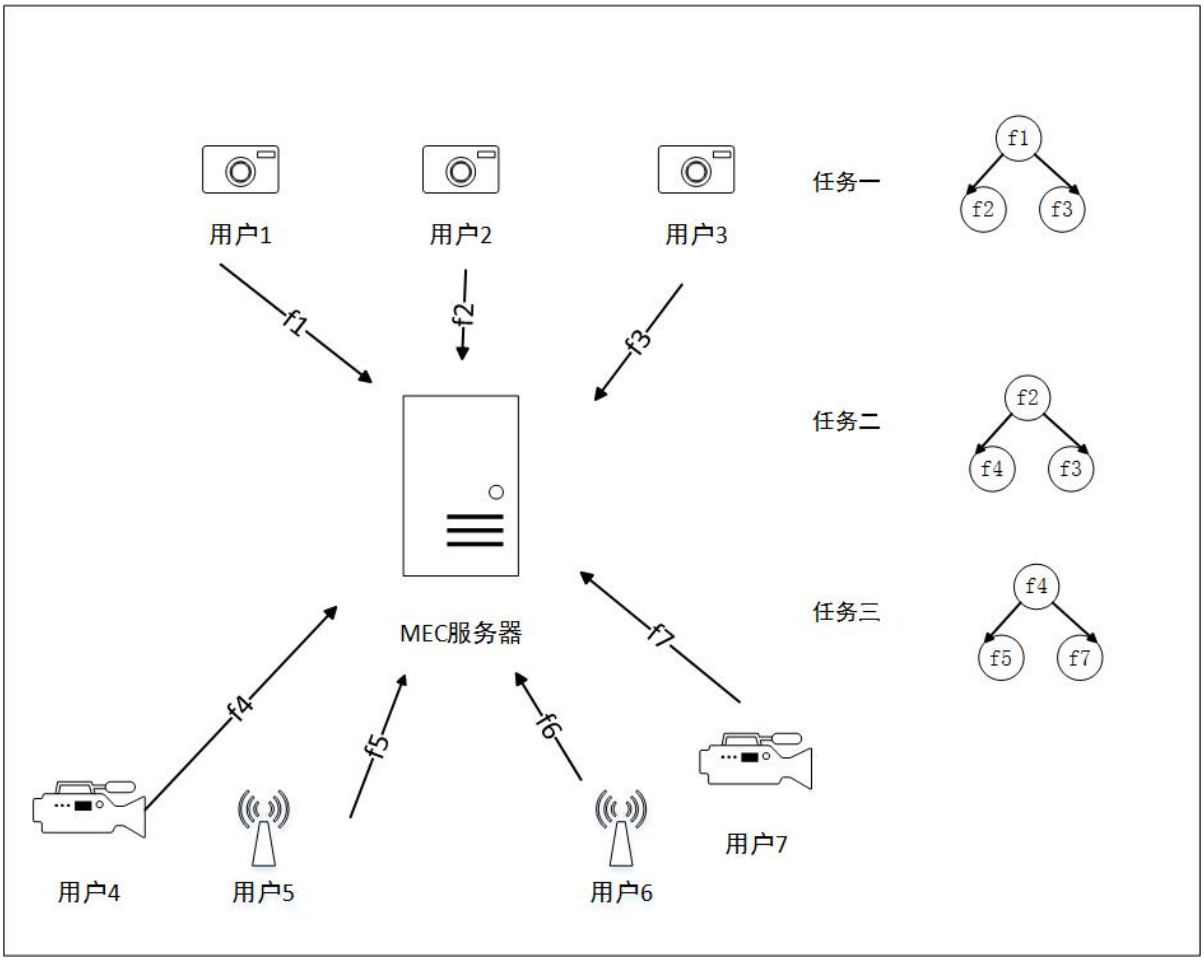

Figure 2. Application scenario diagram
图 2. 应用场景图

$$t_s^m = \begin{cases} \min\left\{t_e^{(m-1)}\right\} & m > 1 \\ 0 & m = 1 \end{cases}$$
$$t_e^m = \left\{t_s^m + \frac{d_i}{B_i}\right\} \tag{1}$$

用 $T_n = \left\{T_{a_1}, T_{a_2}, \cdots, T_{a_n}\right\}$ 表示 n 个任务的对应决策时间，边缘服务器某一时刻 j 接收计算卸载请求的任务集合由 A_j 表示。故由(1)可得多任务决策时间为：

$$T_n = \max\left\{t_e^m\right\} \quad \left(a_n \in A_j\right) \tag{2}$$

因此，优化目标多个任务各自完成时间的平均值为：

$$\overline{T} = \frac{1}{p}\sum_{a_w \subseteq A_j} T_{a_w} \quad \left(p = \left|A_j\right|\right) \tag{3}$$

为保证调度方案可靠性，我们引入带宽约束，F_b 表示并行计算的数据流集合：

$$\sum_{f_i \in F_b} B_i \le B \tag{4}$$

因此，多任务计算卸载方法可以表示为：

$$\min \quad \overline{T} = \frac{1}{P} \sum_{a_w \subseteq A_j} T_{a_w} \qquad \left(p = \left|A_j\right|\right)$$

$$\text{st.} \quad \sum_{f_i \in F_b} B_i \le B$$

4. 一种工业智能边缘计算中基于数据流相关性的任务卸载方法

一种工业智能边缘计算中基于数据流相关性的任务卸载方法，包括步骤：S1：MEC 服务器依据不同任务的决策需求，生成相应的决策树，用户端收集特征数据，将同一任务对应的特征数据流归为一个特征数据流组；S2：用户端选择重复率最高的任务向服务器发送计算卸载请求；S3：服务器按重复率选择调度算法，接收任务所包含的特征数据组；S4：服务器计算任务的累积完成时间，以及计算此时多任务平均完成时间。

具体流程如图 3：

1) 某一周期内用户端收集生产线数据并全部上传至 MEC 服务器上；

2) 服务器依据多任务的决策数据及结果，制定相应的决策树；

3) 将任务涉及到的特征编为一组；

$$a_w = \left\{f_1, f_2, \cdots, f_{q_n}\right\}$$

重复上述步骤，生成所有任务的决策树并整理。

4) 计算任意 K 个任务的特征数据流重复率。用 u 表示。

$$u = \frac{d_q}{\sum_{f_i \subseteq A_j} d_i - d_q}$$

5) 用户端选择重复率最高的任务向服务器发送计算卸载请求；

计算卸载请求为：

$$\left[B_i, d_i, u\right]$$

6) 计算 K 个特征数据流组中各个特征数据流的预期最小传输时间 $\frac{d_i}{B_i}$；

7) 如果重复率高于阈值选择调度算法为重复优先算法，则将选定的 K 个特征数据流组中重复特征数据流按预期最小传输时间由小到大进行排序，依次排入调度表中(满足带宽限制的情况下可并行传输)；

8) 当并行计算的某个特征数据流完成传输，则选择剩余特征数据流中预期最小传输时间最小的特征数据流排入调度表中(满足带宽限制)。若不满足带宽限制则安排预期最小传输时间第二小的特征数据流排入调度表中，以此类推，直到该服务器要接收的多特征数据流组中各个特征数据全部传输完成；

9) 如果重复率低于阈值选择调度算法为 TMF 算法(Topological Sort and Expected Minimum Completion Time First 拓扑排序及预期最小完成时间优先联合算法)，则把服务器即将接收 K 个特征数据流组中的特征，依据决策树模型进行拓扑排序，依次排入调度表中(满足带宽限制的情况下可并行传输)；

10) 当并行传输的某个特征完成传输，则选择剩余特征中拓扑排序靠前的特征排入调度表中。注意当遇到同一位次有多个特征可供选择时，将这些特征中预期最小传输时间的特征数据流排入调度表中(满足带宽限制)。若不满足带宽限制则将该位次下预期最小传输时间第二小的特征数据流排入调度表中，以此类推，直到该服务器把将要接收的多特征数据流组中全部特征数据流传输完成。

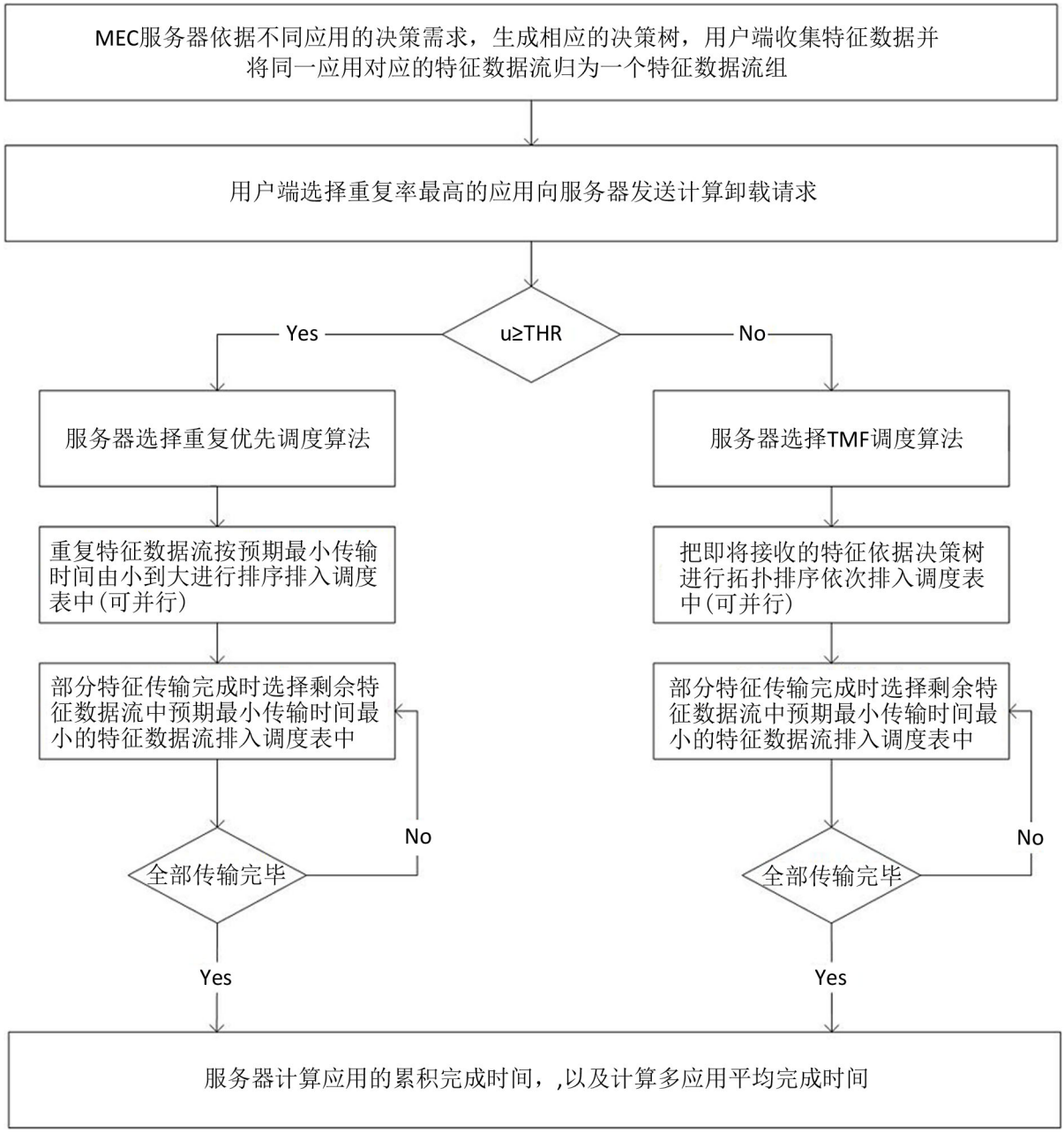

Figure 3. Algorithm flow chart
图 3. 算法流程图

5. 结果评估

用 python 对两种算法进行验证，随机模拟了 10,000 组数据，将重复优先算法和 TMF 算法和传统贪婪算法(预期最小完成时间优先)以及普通卸载进行对比(如图 4)。实验表明(如表 1)面向工业智能边缘计算多任务并存场景下，在服务器计算能力允许的范围内，考虑避免重复数据多次卸载的重复优先算法和 TMF 算法以及预期最小完成时间优先算法因为处理数据量的减少，都比普通卸载时间优化 13.71%以上。

其中，在多任务重复率低于 32.6%时，重复优先算法与预期最小完成时间最优效果相仿，都比随机卸载时间节省 15%~17%。而 TMF 算法比随机卸载时间节省 21.93%。故在多应用重复率较低时，我们可

采用 TMF 算法进行计算卸载。在多任务重复率在 67.1%~92.7%时 TMF 算法与预期最小完成时间最优效果相仿，都比随机卸载时间节省 15%~16%。而重复优先算法效果较优，相比随机卸载时间节省 22.32%。因此，灵活根据重复率选择重复优先算法和 TMF 算法在解决此类问题上具有更高的优化性，满足了工业生产的实时性及可靠性需要。

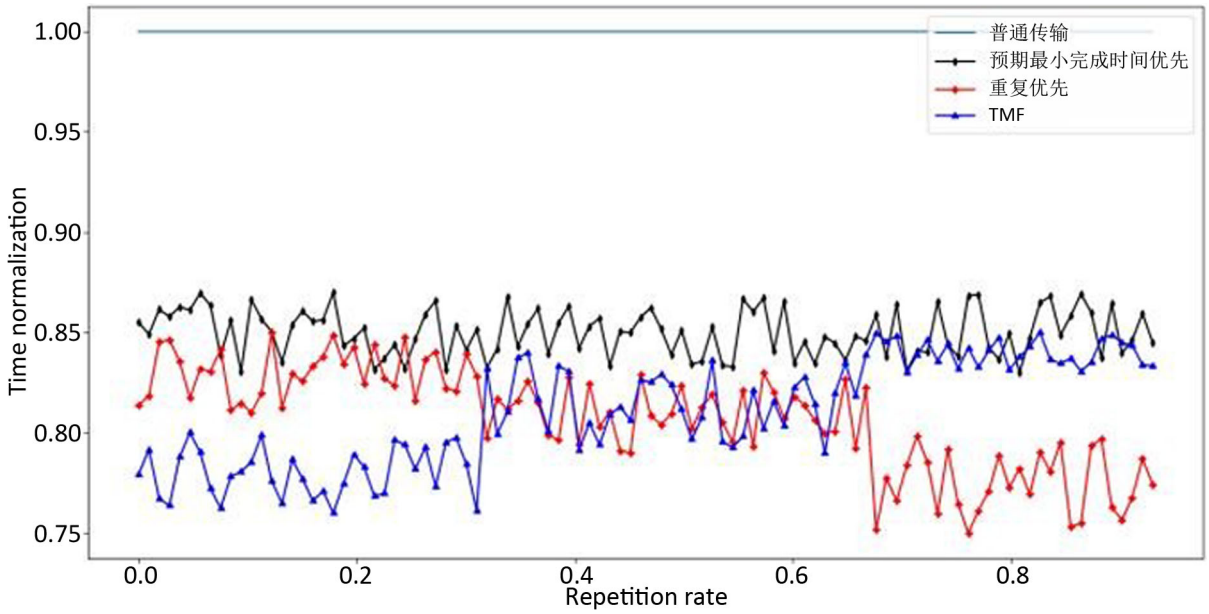

Figure 4. Experimental results
图 4. 实验结果

Table 1. Comparison of algorithms
表 1. 算法比较

算法	最优解数目	平均优化率	预估算法最优区间(重复率)
普通传输	0	0	NULL
预期最小完成时间优先	0.05%	13.71%	NULL
TMF	46.29%	21.93%	0%~32.6%
重复优先	53.66%	22.32%	67.1%~92.7%

6. 结语

本文运用边缘计算和机器学习技术，结合多数据流相关性思想。基于传统贪婪算法(预期最小传输时间优先算法)，提出了一种工业智能边缘计算中基于数据流相关性的计算卸载方法，避免无关数据卸载、多任务场景下重复数据多次卸载及无序卸载导致的缓存过多，存储器容量不足等问题。仿真结果表明该方法相比传统算法具有更高的卸载效率，满足工业场景的实时性需求和可靠性需要。

参考文献

[1] Li, Z., Zhou, X. and Qin, Y. (2019) A Survey of Mobile Edge Computing in the Industrial Internet. 2019 7*th International Conference on Information*, *Communication and Networks* (*ICICN*), Macao, 24-26 April 2019, 94-98. https://doi.org/10.1109/ICICN.2019.8834959

[2] Khan, S., Paul, D., *et al*. (2018) Artificial Intelligence Framework for Smart City Microgrids: State of the Art, Chal-

lenges, and Opportunities. 2018 *Third International Conference on Fog and Mobile Edge Computing* (*FMEC*), Barcelona, 23-26 April 2018, 283-288. https://doi.org/10.1109/FMEC.2018.8364080

[3] Tang, B., Chen, Z., Hefferman, G., *et al*. (2017) Incorporating Intelligence in Fog Computing for Big Data Analysis in Smart Cities. *IEEE Transactions on Industrial Informatics*, **13**, 2140-2150.

[4] Song, Y., Yau, S.S., Yu, R., *et al*. (2017) An Approach to QoS-Based Task Distribution in Edge Computing Networks for IoT Applications. 2017 *IEEE International Conference on Edge Computing* (*EDGE*), Honolulu, 25-30 June 2017, 32-39. https://doi.org/10.1109/IEEE.EDGE.2017.50

[5] Wei, F., Chen, S. and Zou, W. (2018) A Greedy Algorithm for Task Offloading in Mobile Edge Computing System. *China Communications*, **15**, 149-157. https://doi.org/10.1109/CC.2018.8543056

[6] Zhang, D., Ma, Y., Zheng, C., *et al.* () Cooperative-Competitive Task Allocation in Edge Computing for Delay-Sensitive Social Sensing. 2018 *IEEE/ACM Symposium on Edge Computing* (*SEC*), Seattle, 25-27 October 2018, 243-259. https://doi.org/10.1109/SEC.2018.00025

[7] Dab, B., Aitsaadi, N. and Langar, R. (2019) A Novel Joint Offloading and Resource Allocation Scheme for Mobile Edge Computing. 2019 16*th IEEE Annual Consumer Communications & Networking Conference* (*CCNC*), Las Vegas, NV, 11-14 January 2019, 1-2. https://doi.org/10.1109/CCNC.2019.8651879

[8] Liu, C., Cao, Y., Luo, Y., *et al*. (2018) A New Deep Learning-Based Food Recognition System for Dietary Assessment on An Edge Computing Service Infrastructure. *IEEE Transactions on Services Computing*, **11**, 249-261

[9] Ali, M., Anjum, A., Yaseen, M.U., *et al*. (2018) Edge Enhanced Deep Learning System for Large-Scale Video Stream Analytics. 2018 *IEEE* 2*nd International Conference on Fog and Edge Computing* (*ICFEC*), Washington, DC, 1-3 May 2018, 1-10. https://doi.org/10.1109/CFEC.2018.8358733

[10] 黄鸿, 莫李思, 孙罡. 一种基于端口聚合流量的 Coflow 调度机制[J]. 通信技术, 2018, 51(7): 1594-1601.

[11] 马腾, 胡宇翔, 张校辉. 基于深度增强学习的数据中心网络 Coflow 调度机制[J]. 电子学报, 2018, 46(7): 84-91.