DOI: https://doi.org/10.15276/eltecs.44.120.2025.9
УДК 004.93'1:004.8

Комбінована фільтрація даних з датчиків рівня пального на мобільних платформах із застосуванням класичних алгоритмів та LSTM-мережі

Є. В. Тенета, В. С. Ситніков
Національний університет «Одеська політехніка»

Анотація. *У статті розглянуто проблему шумів та інерційних викривлень у сигналах з датчиків рівня пального на транспортних засобах. Запропоновано комбінований підхід, що поєднує класичні методи згладжування (ковзне середнє, медіанна фільтрація) з глибинним навчанням на основі LSTM-мережі. Для формування еталонних даних використано як реальні, так і синтетично згенеровані приклади. Проведено експериментальний аналіз, який показав переваги комбінованої моделі над традиційними алгоритмами за точністю та стабільністю. Результати свідчать про перспективність інтеграції підходу у системи моніторингу транспорту.*

Ключові слова: *датчики рівня пального,мобільні платформи, згладжування сигналів, ковзне середнє, медіанна фільтрація, експонентне згладжування, рекурентна нейронна мережа, LSTM, комбінована фільтрація, навчання з учителем, синтетичні дані.*

Вступ

Сучасні інтелектуальні транспортні системи потребують високоточного моніторингу витрат пального за допомогою датчиків рівня пального (ДРП). Проте дані з таких сенсорів часто спотворюються через шум, інерційні коливання та інші зовнішні впливи, що ускладнює їх коректну обробку, аналіз та використання для прийняття рішень.

Найбільш гостро ця проблема постає в умовах мобільних платформ — вантажних автомобілів, сільськогосподарської та спеціальної техніки, де під час руху коливання пального в баку та зміни положення транспортного засобу призводять до значних флуктуацій показників. Без додаткової обробки сигналу такі вимірювання не забезпечують належної достовірності.

Для підвищення точності показників рівня пального необхідно застосовувати методи фільтрації, здатні ефективно усувати шум та зменшувати вплив інерційних спотворень. У даній роботі запропоновано комбінований підхід, який поєднує класичні алгоритми згладжування (ковзне середнє, медіанна фільтрація) із сучасними методами глибинного навчання — зокрема, рекурентною нейронною мережею типу LSTM.

1 Аналіз проблеми та особливості сигналів з датчиків рівня пального

1.1 Актуальність проблеми

Системи моніторингу транспорту активно використовують датчики рівня пального (ДРП) для контролю витрат пального, обліку заправок, виявлення витоків і шахрайства. Проте на практиці дані, що надходять з ДРП, не завжди придатні для прямого аналізу: вони містять значний рівень шуму, інерційні коливання, артефакти через зміну швидкості або положення транспортного засобу.

Особливо критичними ці викривлення стають у русі — під час гальмування, розгону, поворотів, на ухилах. Через коливання рідини в баку сигнал з ДРП часто не відображає реального рівня пального в конкретний момент часу. Також проблема ускладнюється різкими піками при заправках або витоках, які можуть бути сприйняті як помилка або шум, якщо не застосовувати спеціальну обробку.

У цьому контексті виникає потреба у гнучких та інтелектуальних алгоритмах згладжування, які здатні адаптуватися до реального характеру сигналу і зменшувати вплив збурень.

1.2 Структура сигналу з ДРП

Сигнал з датчика рівня пального зазвичай має вигляд кусково-постійної функції з періодичними змінами, які відображають витрати пального або заправки. Основні типи характерних змін:

– Плавні спадання — природне споживання пального при русі;

– Різкі зростання — заправки;

– Коливання навколо рівня — спричинені інерційними хвилями в баку;

– Випадкові флуктуації — шум електроніки, перешкоди.

Найбільш проблемним є інерційний компонент: при зміні швидкості або нахилу транспортного засобу рівень пального змінюється миттєво, хоча фактичної витрати чи заправки не відбувається. Це створює фальшиві "сигнали", які унеможливлюють просту побудову аналітики без попередньої фільтрації.

1.3 Недоліки традиційних підходів

Прості способи згладжування, як-от ковзне середнє або медіанна фільтрація, хоча й дозволяють зменшити шум, не враховують контекст і не можуть адаптуватися до різних ситуацій — наприклад, відрізнити справжню заправку від флуктуацій.

Крім того, при надмірному згладжуванні можлива втрата важливих характеристик — наприклад, згладжене значення може суттєво відставати від фактичного рівня пального в моменти різкої зміни, особливо при низькому розмірі буфера або вікна фільтра.

Унаслідок цього постає потреба в комбінованих підходах, які об'єднують сильні сторони простих алгоритмів із потенціалом нейромереж, що і є предметом даного дослідження.

2. Методи згладжування даних з ДРП

Для підвищення достовірності сигналів з датчиків рівня пального у роботі було досліджено низку методів згладжування, що охоплюють як класичні алгоритми цифрової обробки, так і сучасні підходи на основі машинного навчання. Зокрема, розглянуто ковзне середнє як базовий інструмент пригнічення шуму, медіанну фільтрацію для усунення поодиноких викидів, експоненційне згладжування (EMA), яке враховує вагу попередніх значень та дозволяє швидше реагувати на зміни, а також рекурентну нейронну мережу LSTM, здатну адаптивне відрізняти істинні події від шумових флуктуацій. Такий спектр методів забезпечує можливість порівняльного аналізу та створює основу для комбінованого підходу до фільтрації даних.

2.1 Ковзне середнє

Ковзне середнє (Moving Average, MA) — це один із найпростіших і найпоширеніших методів цифрової фільтрації. Суть полягає у заміні кожного елемента часового ряду середнім арифметичним значенням із вікна фіксованого розміру, яке "ковзає" по масиву даних[1].

Формула ковзного середнього:

$$\widehat{x_i} = \frac{1}{N}\sum_{j=i-\frac{N-1}{2}}^{i+\frac{N-1}{2}} x_j,$$

де: $\hat{x}_i$— згладжене значення в точці I;

N — розмір вікна згладжування (непарне число);

x_j— початкові виміряні значення.

Переваги:

– Простота реалізації;

– Добре усуває високочастотний шум.

Недоліки:

– При великих вікнах — втрата точності у місцях різких змін (наприклад, заправки);

– Відставання (лаг) у згладженому сигналі.

2.2 Медіанна фільтрація

Медіанна фільтрація (Median Filter) — це нелінійний метод згладжування, який замінює кожне значення на медіану значень у сусідньому вікні. На відміну від середнього, медіанна фільтрація краще зберігає "структуру" сигналу й не розмиває різкі переходи[2].

Формула медіанної фільтрації:

$$x^i = median(xi - k, \dots, xi, \dots, xi + k),$$

де $2k + 1 = N$ — розмір вікна.

Переваги:

– Ефективно прибирає одиничні сплески або падіння;

– Зберігає форму сигналу краще за MA.

Недоліки:

– Нестабільна поведінка в межових значеннях;

– Неможливо задати адаптивність до контексту (наприклад, ігнорувати заправку як "легітимний пік").

2.3 Експонентне згладжування (EMA)

Експонентне згладжування (Exponential Moving Average, EMA) — це рекурсивний фільтр першого порядку, що надає більшій вазі свіжим спостереженням і тим самим зменшує лаг порівняно з ковзним середнім [3]. Базове рівняння має вигляд

$$y_i = \alpha x_i + (1 - \alpha)\, y_{i-1}, \quad 0 < \alpha \leq 1,$$

де: x_i — сирий сигнал;

y_i — згладжене значення,

α — коефіцієнт згладжування (чим більше α, тим швидша реакція і менше згладжування) [3]. Для задач із фіксованим кроком дискретиза

ції Δt зручно задавати α через часову константу τ:

$$\alpha = 1 - e^{-\Delta t/\tau}.$$

У роботі використано також двонаправлене (двопрохідне) EMA для зменшення фазового зсуву: пряме згладжування виконується зліва направо, зворотне — справа наліво, після чого результати усереднюються:

$$\tilde{y}_i = 1/2\left(y_i^{\text{fwd}} + y_i^{\text{bwd}}\right).$$

Переваги: менший лаг, ніж у MA; простота й обчислювальна ефективність (онлайн-режим); плавна реакція на тренд зі збереженням локальної структури.

Обмеження: вибір α критично впливає на компроміс «шум ↔ зсув»; як і інші згладжувачі, EMA може "розмазувати" різкі події (заправки), якщо їх не маскувати; потребує коректної ініціалізації y_0 (наприклад, першим валідним значенням або середнім по короткому вікну).

Практичні налаштування для ДРП: (i) застосовувати попередню обробку — заміну нулів останнім дійсним значенням та маскування зон заправок; (ii) добирати α або τ під динаміку платформи: $\alpha \approx 0.05 - 0.15$ для «спокійних» профілів і $\alpha \approx 0.2 - 0.35$ для більш шумних умов; (iii) для офлайн-аналізу віддавати перевагу двонапрямному EMA (нульова фаза), для онлайн-інференсу — каузальному варіанту з чітко визначеним лагом.

2.4 Рекурентна нейромережа LSTM

LSTM (LongShort-TermMemory) — це тип рекурентної нейронної мережі (RNN), спеціально розроблений для роботи з послідовностями даних[4]. LSTM зберігає інформацію про довгострокові залежності, що дозволяє їй ефективно передбачати майбутні або очищені значення на основі контексту.

Застосування в даному проекті:

– Модель навчається на парах "сирий сигнал – згладжене значення", отриманих із класичних фільтрів (MA + медіанний);

– На вході — послідовність N значень рівня пального (наприклад, 15 або 21 точка);

– На виході — згладжене значення в центрі вікна.

Переваги:

- Модель "вчиться" реальному характеру сигналу;

– Можливість адаптації до різних типів шуму, амплітуд, транспортних засобів.

Недоліки:

– Вимагає попередньої підготовки даних (еталонів);

– Потрібне навчання і контроль якості;

– Повільніше, ніж класичні фільтри.

Таким чином, LSTM-мережа працює як мета-фільтр, що імітує розумну поведінку згладжування, навчаючись на добре підібраних еталонах з класичних алгоритмів.

3 Практична реалізація комбінованої фільтрації

У межах дослідження реалізовано користувацький вебінтерфейс для генерації, візуалізації та обробки часових рядів рівня пального, який інтегрує класичні алгоритми фільтрації (ковзне середнє, медіанна фільтрація, експоненційне згладжування) та допоміжні процедури попередньої обробки (ігнорування зон заправок, заміна нульових значень, придушення дрібних підйомів). Інтерфейс підтримує ручне коригування та побудову контрольної лінії й забезпечує збереження узгоджених пар «raw→expected» у стандартизованому форматі для подальшого навчання. На серверній стороні розгорнуто конвеєр навчання нейромережі типу LSTM на сформованих еталонах і сервіс інференсу, що надає згладжування нових даних у реальному часі через REST-API. Така архітектура поєднує прозоре формування високоякісних таргетів із відтворюваним навчанням моделі та дозволяє оперативно інтегрувати результати у виробничі системи моніторингу.

3.1 Підготовка навчальних даних

Навчання нейромережевої моделі типу LSTM проводилося у режимі «з учителем», що вимагало формування великої кількості пар даних формату raw → expected. Для цього було реалізовано кілька допоміжних інструментів і процедур попередньої обробки, які суттєво покращили якість навчального набору та дозволили уникнути помилкових патернів у моделі.

3.1.1 Виявлення та пропуск заправок. Медіанне згладжування — це ефективний метод усунення імпульсних шумів у часових рядах. Для кожної точки ряду x_i, значення замінюється медіаною сусідніх значень у вікні шириною w:

$$\hat{x}_i = \text{median}(x_{i-k}, \dots, x_i, \dots, x_{i+k}),$$

де $w = 2k + 1$.

Однак при наявності різких справжніх підйомів (наприклад, заправок пального), медіанне згладжування може некоректно "затирати" ці ділянки, знижуючи точність моделі.

Тому було реалізовано автоматичне виявлення та пропуск заправок, що враховує наступні умови:

– Підйом вгору $\Delta x > \delta_{\text{refill}}$, де δ_{refill} — порогове значення (наприклад, 4.0 літри);

– Зона навколо такого стрибка шириною r точок в кожен бік вважається refill-зоною і не згладжується.

Алгоритм для ігнорування заправок:

– Проходимо дані і виявляємо стрибки $x_i - x_{i-1} > \delta_{\text{refill}}$

– Для кожного такого індексу додаємо у множину R всі індекси від $i - r$ до $i + r$

– Під час згладжування, якщо $i \in R$, залишаємо $\hat{x}_i = x_i$ без змін

Це дозволяє зберігати справжні стрибки (заправки) і водночас ефективно пригнічувати інші імпульсні коливання.

На практиці, це згладжування реалізоване у вигляді кнопки у інтерфейсі "📐 Медіанне згладжування", з опцією "⛽ ігнорувати заправки", яку користувач може вмикати за потреби. Це значно покращує якість попередньої обробки даних перед подачею до нейронної мережі.

Метод ковзного середнього є класичним підходом до зменшення шуму у часових рядах. Для кожного елементу ряду x_i обчислюється середнє значення вікна шириною w, що охоплює $w = 2k + 1$ точок:

$$\hat{x}_i = \frac{1}{w}\sum_{j=i-k}^{i+k} x_j.$$

Цей метод є простим, ефективним і часто використовується як базова техніка згладжування.

Проте при аналізі рівня пального цей підхід також має суттєвий недолік — він спотворює справжні стрибки, спричинені заправками. Щоб уникнути цього, було додано логіку автоматичного виявлення таких зон, аналогічну до тієї, що використовувалась у медіанному згладжуванні.

Умови виявлення заправки:

– Стрибок $x_i - x_{i-1} > \delta_{\text{refill}}$, де δ_{refill} — заданий поріг (наприклад, 4.0 л);

– Навколо такого стрибка визначається зона шириною r точок (радіус), яка вважається заправкою і виключається зі згладжування.

Таким чином, фільтрація виконується за формулою:

$$\hat{x}_i = \begin{cases} x_i, & \text{якщо } i \in R \\ \dfrac{1}{w}\displaystyle\sum_{j=i-k}^{i+k} x_j, & \text{інакше} \end{cases}$$

де R — множина індексів точок, що потрапляють у зони заправок.

Цей підхід дозволяє:

– Зберігати справжні підйоми, пов'язані із заправками;

– Ефективно пригнічувати випадкові флуктуації рівня пального;

– Уникати надмірного згладжування важливих ділянок сигналу.

Функціональність реалізована у вебінтерфейсі як кнопка "🧹 Згладжування", з можливістю активувати прапорець "ігнорувати заправки ⛽". Це дає змогу адаптувати обробку до контексту даних та уникнути втрати важливої інформації перед подачею до алгоритмів виявлення або навчання моделі.

3.1.2 Заміна нульових значень останнім дійсним. У сирих даних часто трапляються обнуління значення рівня пального — через обрив з'єднання, помилку зчитування або нестабільність електроживлення датчика. Такі нулі спотворюють картину сигналу і призводять до хибного навчання.

Тому у ході попередньої обробки даних перед згладжуванням було реалізовано механізм заміни нульових значень (fl = 0) останнім дійсним (ненульовим) значенням. Це дозволило уникнути хибних провалів у сигналі, які виникали внаслідок короткочасних помилок зчитування або перебоїв у живленні датчика рівня пального.

Математично ця операція описується так. Нехай маємо часовий ряд значень рівня пального:

$$X = \{x_1, x_2, x_3, \dots, x_n\}, \quad x_i \in R_{\geq 0}.$$

Після обробки отримуємо перетворений ряд

$$\hat{X} = \{\widehat{x_1}, \widehat{x_2}, \dots, \widehat{x_n}\},$$

де $\hat{x}_i =$

$$\begin{cases} x_i, & x_i \neq 0 \\ \hat{x}_{i-1}, & x_i = 0 \text{ і } \hat{x}_{i-1} \neq 0 \\ 0, & x_i = 0 \text{ і попередніх валідних значень немає} \end{cases}$$

Цей метод дозволяє уникнути появи штучних збоїв у сигналі та зберегти неперервність, що особливо важливо для нейромережевого навчання на основі віконної розбивки. У інтерфейсі цей інструмент реалізовано як окрему кнопку "Заміна нулів", яка дозволяє швидко застосувати обробку до завантаженого графіка.

3.1.3 Фільтрація дрібних стрибків рівня пального. Рівень пального в реальних умовах може коливатися через вібрації, рух транспорту або інші випадкові фактори, що створює хибні короткі підйоми (falsepositive) у даних. Для підвищення якості навчання нейромережі було реалізовано фільтр, який блокує малі за величиною

стрибки вгору, якщо їх амплітуда не перевищує заданий поріг $\Delta_{\max}$.

Алгоритм працює покроково для кожного елементу часового ряду x_i, порівнюючи його з попереднім:

$$\hat{x}_i = \begin{cases} x_i, & \text{якщо } x_i - \hat{x}_{i-1} \geq \Delta_{\max} \\ \hat{x}_{i-1}, & \text{якщо } 0 < x_i - \hat{x}_{i-1} < \Delta_{\max} \\ x_i, & \text{якщо } x_i \leq \hat{x}_{i-1} \end{cases}$$

де: $\hat{x}_i$ — скориговане значення рівня пального на кроці i;

x_i — початкове (сире) значення на кроці i;

$\Delta_{\max}$ — допустимий поріг росту (типово 2.0 літри).

Ця обробка не впливає на падіння рівня пального, лише на штучні стрибки вгору, що не є заправками. На практиці це реалізовано як кнопка «💾 Фільтр росту» у інтерфейсі, що дозволяє швидко очистити сигнал перед згладжуванням або подачею в нейромережу.

3.1.4 Ручна корекція точок згладжування. Після застосування класичних фільтрів (ковзне середнє або медіанна фільтрація), користувач мав змогу вручну підправити сигнал у вебінтерфейсі. Це було необхідно в наступних випадках:

– якщо фільтр занадто "зрізав" пік заправки;

– якщо була помилка в зоні шуму;

– якщо треба було відновити горизонтальний рівень (плато), зруйноване фільтрацією.

Таким чином, еталонна лінія expected ставала високоякісною та наближеною до ідеального рішення, яке модель мала би навчитися імітувати.

3.1.5 Генерація синтетичних графіків для навчання. Для забезпечення достатнього обсягу навчальних прикладів була реалізована система створення псевдореалістичних «шумних» графіків рівня пального. На відміну від повної автоматичної генерації пар raw/expected, у даному підході автоматично формується лише сирий (шумлений) графік, а його еталонна версія (expected) створюється вручну або з використанням доступних фільтрів безпосередньо у вебінтерфейсі.

Такий підхід дозволяє:

– швидко отримувати різноманітні вихідні криві для навчання моделі з учителем;

– повністю контролювати сценарії споживання пального та рівень шуму;

– редагувати та коригувати дані з урахуванням контексту (наприклад, ігнорування заправок або корекція нульових значень).

Кожен згенерований графік є часовим рядом з n точок, де:

– t_i — мітка часу (Unix-час);

– fl_i — рівень пального в літрах у момент t_i.

Параметри генерації:

– Початковий рівень пального $F_0 \in [20, 100]$ л.

– Кінцевий рівень $F_1 = F_0 \times (1 - u)$, де $u \in [0.3, 1.0]$ — частка витрати.

– Рівномірне споживання з випадковими флуктуаціями (±20% до кожного кроку).

– Додатковий випадковий шум $\eta_i \in [-\varepsilon, +\varepsilon]$, де $\varepsilon = F_0 \times 0.04$.

Базова формула:

$$fl_i = \max(0, F_0 - i \cdot \delta_i + \eta_i)$$
$$\delta_i = (F_0 - F_1) / n \cdot (1 + \varepsilon_i)$$
$$\varepsilon_i \in [-0.2, 0.2], \ \eta_i \in [-\varepsilon, \varepsilon]$$

Особливості генерації:

– Заправка: з ймовірністю 30% додається подія «доливу» обсягом $r \in [F_0 \times 0.05, F_0 \times 0.9]$ на певному кроці;

– Стоянка: з ймовірністю 50% формується «плато» — ділянка без споживання, де шум зменшується у 6 разів;

– Таймінг: інтервал між точками — 60 секунд, початкова дата — випадкова в межах 2022–2025 років.

Процес формування пар raw/expected:

– Автоматично: генерується шумлений графік (raw) з реалістичними сценаріями (споживання, стоянки, заправки, шум);

– Напівавтоматично або вручну: у вебінтерфейсі користувач застосовує фільтри (ковзне середнє, медіанна фільтрація, згладжування з ігноруванням заправок) або редагує точки вручну;

– Збереження пари: у результаті формується навчальна пара — початковий «шумний» графік і очищений еталон (expected), які зберігаються з унікальним ідентифікатором.

Переваги такого підходу:

– Можливість контролювати якість таргету перед додаванням у навчальний набір.

– Врахування контекстних особливостей конкретного сценарію (наприклад, часткові заправки, нетипові коливання).

– Гнучке створення унікальних прикладів як для стандартних, так і для рідкісних випадків (міське таксі, сільгосптехніка, далекобійні перевезення).

– Узгодженість між raw і expected завдяки однаковим алгоритмам обробки.

Таким чином, навіть без повної автоматизації формування навчальних пар підхід забезпечує

високу достовірність даних, можливість адаптації під конкретні умови та більш ефективне навчання LSTM-моделі у режимі «навчання з учителем».

3.2 Розробка редактора графіка на базі вебінтерфейсу

Для зручного створення та візуального редагування еталонних даних розроблено вебінтерфейс на базі HTML, JavaScript та бібліотеки ECharts. Головні функції редактора:

– імпорт JSON-файлів (локально або за URL);

– генерація синтетичних даних;

– відображення трьох графіків: «оригінальний», «редагований», «контрольна лінія»;

– згладжування даних через кнопки: ковзне середнє, медіанна фільтрація, експоненційне згладжування (EMA) з параметром α;

– двонапрямне (двопрохідне) EMA для зменшення фазового зсуву;

– опції попередньої обробки: ігнорування зон заправок, заміна нульових значень останнім дійсним, фільтр дрібних підйомів;

– ручне коригування червоних точок і прив'язка до зеленої контрольної лінії;

– створення снапшота (початкових даних) і збереження результатів;

– експорт узгоджених пар у форматі {timestamp}.json і {timestamp}e.json;

– виклик інференсу через FastAPI (/smooth) і згладжування нейромережею;

– відправлення розмічених даних на навчання (Submitfor Training) і запуск донавчання моделі (Retrain Model);

– відкриття/експорт SVG та скидання стану (Reset).

Цей інструмент дозволив швидко формувати якісні еталони для навчання моделі та аналізу фільтраційних методів. На рис. 1 наведено основні елементи вебінтерфейсу.

3.3 Реалізація нейромережі LSTM

3.3.1 Постановка задачі. Завдання формулюється як регресія у режимі навчання з учителем: для кожної точки шумного ряду $x_1, \dots, x_T$ необхідно оцінити «очищене» значення y_t (еталон, отриманий ручною/напівавтоматичною обробкою). Модель навчається апроксимувати відображення

$$f_\theta\colon \mathbb{R}^L \to \mathbb{R}, \qquad \hat{y}_t = f_\theta(W_t),$$

де W_t — вікно навколо індексу t довжини L=2k+1.

У реалізації використано $k = 5$ (тобто $L = 11$). Для країв ряду застосовано реплікаційне доповнення (edge-replication), щоб зберегти розмір вікна:

$$W_t = [x_{t-k'}, \dots, x_t, \dots, x_{t+k''}],$$

де $\begin{cases} k' = \min(k, t-1), \\ k'' = \min(k, T-t), \end{cases}$

а відсутні елементи зліва/справа копіюються з крайніх доступних значень $x_{t-k'}$ або $x_{t+k''}$.Ціль формування вибірки:

$$(W_t, y_t), \quad y_t = s_t,$$

де s_t — згладжене/скориговане еталонне значення (створене вручну або за допомогою фільтрів у редакторі).

Зауваження про позначення N: у порівняльних таблицях для класичних фільтрів N — розмір вікна згладжування. У нейромережі ми позначаємо $L = 2k + 1$. У наведеній реалізації $L = 11$ (не 21).

3.3.2 Нормалізація. Застосовано глобальну мін-макс нормалізацію по всьому тренувальному набору:

$$\tilde{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}, \qquad \tilde{y} = \frac{y - y_{\min}}{y_{\max} - y_{\min}}.$$

Параметри $(x_{\min}, x_{\max}, y_{\min}, y_{\max})$ зберігаються у normalization.json для подальшої денормалізації під час інференсу.

3.3.3 Архітектура моделі. Мережу реалізовано у Tensor Flow/Keras [5]:

$$\begin{aligned} &\text{Input}(L \times 1) \;\to\; \text{LSTM}(32) \\ &\quad\to\; \text{Dense}(16,\ \text{ReLU}) \\ &\quad\to\; \text{Dense}(1). \end{aligned}$$

LSTM обробляє послідовність $(\tilde{x}_{t-k}, \dots, \tilde{x}_{t+k})$, формуючи прихований стан h, після чого дві полнозв'язні ланки дають скалярну оцінку $\hat{y}_t$.

Для повноти, динаміка однієї LSTM-комірки:

$$\begin{aligned} \mathbf{i}_\tau &= \sigma(\mathbf{W}_i[\mathbf{h}_{\tau-1}, \mathbf{x}_\tau] + \mathbf{b}_i), \\ \mathbf{f}_\tau &= \sigma\big(\mathbf{W}_f[\mathbf{h}_{\tau-1}, \mathbf{x}_\tau] + \mathbf{b}_f\big), \\ \mathbf{o}_\tau &= \sigma(\mathbf{W}_o[\mathbf{h}_{\tau-1}, \mathbf{x}_\tau] + \mathbf{b}_o), \\ \tilde{\mathbf{c}}_\tau &= \tanh(\mathbf{W}_c[\mathbf{h}_{\tau-1}, \mathbf{x}_\tau] + \mathbf{b}_c), \\ \mathbf{c}_\tau &= \mathbf{f}_\tau \odot \mathbf{c}_{\tau-1} + \mathbf{i}_\tau \odot \tilde{\mathbf{c}}_\tau\,, \\ \mathbf{h}_\tau &= \mathbf{o}_\tau \odot \tanh(\mathbf{c}_\tau), \end{aligned}$$

де σ — сигмоїда, $\odot$ — покомпонентне множення.

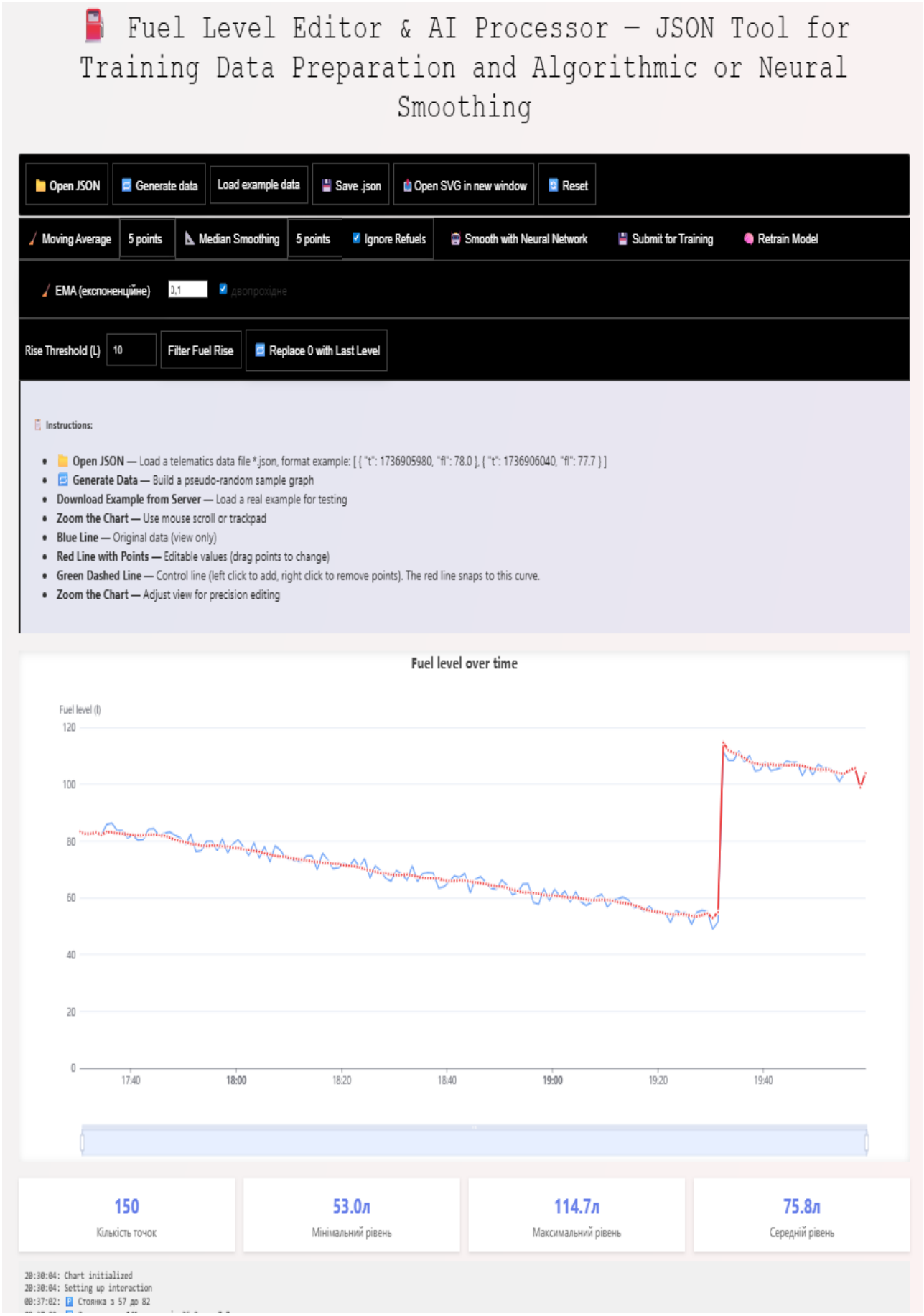

Рис. 1 - Основні елементи вебінтерфейсу

3.3.4 Функція втрат та метрики. Модель оптимізує MSE, а для моніторингу використовується MAE:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n}\sum_{t=1}^{n}(\hat{y}_t - y_t)^2,$$

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|\hat{y}_t - y_t|.$$

3.3.5 Протокол навчання.

Постановка завдання. Модель типу LSTM навчається у постановці регресії за парами $(\text{W}_\text{t}, \text{y}_\text{t})$, де W_t — локальний контекст сирого сигналу, сформований як центроване вікно довжини L навколо індексу t, а y_t — еталонне згладжене значення. Така постановка дозволяє моделі опановувати як локальну динаміку, так і характерні шаблони шуму, притаманні сигналам ДРП, зберігаючи при цьому відтворюваність експериментів і коректність порівнянь з класичними методами згладжування.

Формування вибірки. Вибірка конструюється зі зсувом stride = 1. Для кожної позиції t формується вектор

$$\text{W}_\text{t} = (\text{x}_{\text{t}-\text{k}}, \dots, \text{x}_\text{t}, \dots, \text{x}_{\text{t}+\text{k}}), \qquad \text{L} = 2\text{k} + 1,$$

тобто використовується центроване вікно фіксованої довжини. На межах ряду застосовується реплікаційне доповнення (edgereplication): $\text{x}_{\text{t}<0} = \text{x}_0$, $\text{x}_{\text{t}>N-1} = \text{x}_{\text{N}-1}$, що забезпечує сталий розмір W_t на всій довжині послідовності. Перед побудовою пар перевіряється коректність часової шкали (монотонність позначок часу, відсутність дублікатів).

Попередня обробка. Дані підлягають очищенню: приклади з невалідними значеннями (NaN/Inf) або з порушеннями фізичних обмежень (некоректні часові індекси, негативні рівні) вилучаються або маскуються. Така процедура зменшує ймовірність перенесення артефактів у навчальні таргети й стабілізує збіжність під час оптимізації.

Розбиття вибірки. Для оцінювання узагальнювальної здатності дані діляться на тренувальну та тестову підмножини у пропорції 90/10. Встановлення фіксованого зерна випадковості (random_state = 42) гарантує відтворюваність усіх етапів — від формування сплітів до підрахунку метрик.

Процедура навчання. Мережа оптимізується алгоритмом Adam з гіперпараметрами epochs = 150 та batch_size = 32[6]. В якості функції втрат використовується MSE, як базову метрику контролю — MAE (за потреби додатково оцінюється RMSE). Така конфігурація забезпечує стійку збіжність і чутливість до характерних для ДРП локальних відхилень.

Збереження артефактів. Після навчання ваги моделі фіксуються у файлі fuel_lstm_model.h5. Паралельно зберігаються параметри глобальної мін–макс нормалізації у normalization.json (значення $\text{y}_{\min}, \text{y}_{\max}$), що необхідні для коректного переходу між нормованим простором моделі та фізичною шкалою під час інференсу.

Режими інференсу. Оцінювання нового ряду $\{\text{x}_\text{t}\}_{\text{t}=0}^{\text{N}-1}$ виконується ковзним вікном довжини L по всій послідовності; для перших та останніх $\text{k} = (\text{L}-1)/2$ позицій застосовується та сама реплікаційна схема, що й при формуванні навчальних прикладів. Нормований вихід $\hat{\text{y}}_{\text{norm}}$ перетворюється у фізичну шкалу за правилом

$$\hat{\text{y}} = \hat{\text{y}}_{\text{norm}}\,(\text{y}_{\max} - \text{y}_{\min}) + \text{y}_{\min}.$$

Окрім центрованого режиму, для потокових сценаріїв передбачено каузальний інференс без «погляду в майбутнє» із вікном $[\text{t}-\text{L}+1, \dots, \text{t}]$ та таргетом y_t; у цьому випадку лаг інтерпретується як технологічна плата за каузальність.

Узгодженість з експериментами та порівнянність. Порівняльні таблиці для класичних фільтрів (MA/Median) сформовано при розмірі вікна $\text{N} = 9$, тоді як у LSTM-моделі типовим є $\text{L} = 11$. Для коректного бенчмарку доцільно виконати грід-пошук $\text{L} \in \{9,11,13\}$ і зіставляти результати з класичними методами при відповідних розмірах вікон. Базовим є центрований режим, а для реального часу рекомендується каузальний, із відповідним переформулюванням таргетів та інтерпретацією затримки.

Опціоні розширення. Стабільність навчання підвищується застосуванням Early Stopping та Reduce LROnPlateau; для зниження ризику перенавчання доцільно використовувати Dropout (0.2–0.3) після LSTM-шару. Як альтернативні архітектури розглядаються Bidirectional LSTM та GRU (32–64 прихованих одиниць). Підвищенню інформативності входу сприяє розширення вікна похідними ознаками — приріст $\Delta\text{x}_\text{t} = \text{x}_\text{t} - \text{x}_{\text{t}-1}$, локальний ковзний нахил/тренд, бінарні маркери подій «refill/stop». Для неупередженої оцінки на часових рядах варто віддавати перевагу кросвалідації з розбиттям типу Time Series Split замість випадкового спліту.

3.4 Інтеграція через API

Нейромережеву модель було обгорнуто у веб-сервер на базі FastAPI, який забезпечує інтерфейс для обробки даних у режимі реального

часу[7]. Основний ендпоінт / smooth приймає POST-запит, що містить масив точок у форматі JSON (час та рівень пального) та додаткові параметри.

У відповідь сервер повертає згладжений масив того ж формату, з урахуванням обраної моделі згладжування (нейромережевої або класичної).

Для захисту доступу до API було реалізовано кілька рівнів перевірки:

– HTTP Referer — допускаються лише запити з дозволеного джерела https://teneta.biz/fuellevel, що захищає від сторонніх сайтів;

– CORS — доступ дозволений лише для вибраних доменів;

– Пароль авторизації — при кожному відправленні даних у запиті має бути вказаний унікальний пароль, який перевіряється на сервері.

У разі невірного пароля або відсутності належного заголовка referer, сервер повертає помилку з кодом 403 Forbidden.

Це забезпечує безпечне використання API виключно з авторизованого клієнтського інтерфейсу, запобігає несанкціонованому доступу та гарантує контроль над запитами до нейромережевої моделі.

3.5 Інструменти та середовище

Проект реалізовано із залученням повного технологічного стеку — від операційних систем і віртуалізаційної інфраструктури до бібліотек для обробки даних, глибинного навчання та інтерактивної візуалізації.

Операційні системи та інфраструктура:

– Ubuntu Server (на VMwareESXi) — основне серверне середовище розгортання моделей і REST-API;

– macOS (MacBookAir M1) — клієнтське середовище розробки, тестування та підготовки експериментів;

– VMwareESXi — віртуалізована платформа для хостингу сервісів і ізоляції обчислювальних ресурсів.

Мови програмування:

– Python — бекенд, підготовка даних, навчання та інференс нейронної мережі;

– абсолютна похибка між згладженим сигналом та еталонним значенням[8];

– RMSE (Root Mean Square Error) — середньоквадратична похибка, чутливіша до одиничних великих відхилень[9];

– Δmax — максимальне абсолютне відхилення згладженого сигналу від еталонного;

– JavaScript — клієнтський інтерфейс (frontend) і реалізація частини алгоритмів у браузері;

– HTML/CSS — структура та оформлення вебінтерфейсу редактора.

Бібліотеки й фреймворки:

– TensorFlow (Python) — створення, навчання та використання LSTM-моделі;

– FastAPI (Python) — RESTful API для онлайн-згладжування та сервісів інференсу;

– NumPy, Pandas — підготовка вибірок вікон, обробка та аналітика часових рядів;

– ECharts (Java Script) — інтерактивна побудова та редагування графіків рівня пального в браузері;

– Plotly (опційно) — додаткова візуалізація результатів у звітах і під час тестування;

– JSON — уніфікований формат обміну між компонентами системи.

Середовища розробки та інструменти:

– Visual Studio Code — основне IDE для frontend і backend;

– Tensor Board — моніторинг процесу навчання та аналіз метрик;

– Postman / curl — ручне тестування API-ендпоінтів;

– Git — керування версіями коду та відтворюваність експериментів;

– Docker (опційно) — контейнеризація сервісів і ізоляція оточення під час розгортання.

Таким чином, реалізація охоплює повний цикл: від браузерного збору й інтерактивної візуалізації даних до серверної нейромережевої обробки з мінімальною затримкою, чітким розмежуванням відповідальностей компонентів і можливістю гнучкої кастомізації.

4. Результати фільтрації та порівняльний аналіз методів

4.1 Методологія оцінювання якості згладжування

Для об'єктивної оцінки ефективності застосованих методів фільтрації було використано кілька ключових метрик:

– MAE (Mean Absolute Error) — середня

– Lag (затримка) — усереднена зміщеність пікових/переломних точок у часі після фільтрації, порівняно з еталонним графіком.

Порівняння проводилось на незалежному тестовому наборі часових рядів. Всі методи (ковзне середнє, медіанне згладжування, LSTM) використовувались на одних і тих самих фрагментах даних із однаковими параметрами.

4.2 Візуальний аналіз результатів

На численних прикладах із реальних добових записів з ДРП було побудовано графіки:

– Ковзне середнє пригнічує шум, але створює лаг і «розмазує» піки;

– Медіанна фільтрація зберігає форму піків, але створює рвану структуру на плато;

– EMA більш плавний, але «легковажний», що може спричинити помилки;

– LSTM згладжує розумно: залишає заправки, пригнічує шум, не створює лагу.

4.3 Кількісні результати тестування

Кількісні результати тестування наведено в таблиці 1. Порівнюються методи ковзного середнього, медіанної фільтрації, EMA і нейромережева модель LSTM. Оцінювання виконано на відкладеній тестовій вибірці; як метрики використано MAE, RMSE, $\Delta_{\max}$ та лаг (у точках). Менші значення MAE/RMSE/$\Delta_{\max}$ відповідають кращій якості згладжування, для лагу — меншій затримці.

LSTM показує найменші значення похибок і затримки, особливо у випадках заправок, шуму та плато.

Таблиця 1

Кількісні результати тестування

Метод	MAE	RMSE	Δmax	Lag (точок)
Ковзне середнє (N=9)	0.73	0.95	2.2	≈2
Медіанна фільтрація (N=9)	0.68	0.88	1.9	≈1
EMA	0.6	0.7	1.3	≈1
LSTM (навчена модель)	0.42	0.57	1.1	<1

Висновки

Отримані результати дослідження чітко демонструють, що комбінований підхід до підготовки еталонних даних (таргетів), який включає ковзне середнє (MA), медіанну фільтрацію, заміну нульових значень, а також виключення зон заправок, є критично важливим етапом перед навчанням нейронної мережі. Саме такий підхід дозволив отримати чисті, фізично обґрунтовані й логічно послідовні ряди, які в подальшому були використані як навчальні приклади для LSTM-моделі.

Особливо важливо підкреслити, що навчання з учителем (supervisedlearning), яке було застосоване у цьому дослідженні, є основою побудови сучасних високоточних моделей у промисловості. У процесі supervisedlearning кожен вхідний приклад (шумні вимірювання з датчика) супроводжується правильною відповіддю (еталонним значенням рівня пального). Такий формат дозволяє алгоритму чітко зрозуміти, як слід реагувати на різні патерни даних — від різких стрибків і спадів до стабільних плато.

Методика, що використана в роботі, безпосередньо перекликається з підходами, які застосовують великі корпорації — наприклад, Google у своїй системі reCAPTCHA, де мільярди користувачів по суті виконують роль «вчителя», маркуючи дані для тренування моделей комп'ютерного зору. Там, як і в нашому випадку, якість та чистота навчального набору безпосередньо визначає точність та стабільність роботи моделі в реальних умовах.

У випадку з даними рівня пального попередня обробка — це не просто бажаний етап, а абсолютна необхідність. Ситуації, які виникають на мобільних платформах, надзвичайно різноманітні:

– транспорт може перебувати на ухилі або гальмувати, що створює інерційні коливання;

– можливі часткові заправки, що відрізняються від стандартного «стрибка» рівня;

– у сільськогосподарській техніці додатково впливають вібрації та нерівності ґрунту;

– у міському таксі часто відбуваються короткі простої з включеним двигуном, що також відбивається на рівні пального;

– у логістичних фур можливі великі періоди стабільності з різкими заправками, які слід правильно ідентифікувати.

Без ретельної попередньої обробки та формування лінійфікованих (очищених, узгоджених за структурою) навчальних даних жодна нейромережа, навіть найсучасніша, не зможе стабільно працювати у всіх перелічених сценаріях. Унікальні комбінації шумів і спотворень, які виникають у реальному транспорті, вимагають, щоб модель не просто «знала загальні закономірності», а була навчена на реальних і синтетично згенерованих прикладах, які охоплюють весь спектр можливих умов.

Таким чином, проведений аналіз підтвердив, що:

– LSTM-модель при наявності якісно сформованих таргетів перевершує класичні методи згладжування як за точністю (MAE, RMSE, Δmax), так і за мінімальною затримкою (Lag).

– Комбіноване згладжування та ручне доопрацювання таргетів забезпечує дані, що найбі-

льше наближені до фізичної реальності, уникаючи штучних спотворень.

– Навчання з учителем є ключовим фактором, що визначає здатність моделі адаптуватися до різних типів мобільних платформ — від таксі до сільськогосподарської техніки та далекобійних перевезень.

– Попередня обробка даних — це не допоміжний, а обов'язковий етап, без якого нейромережа не зможе показати стабільний результат у складних та нестандартних сценаріях.

Підсумовуючи, можна стверджувати, що розроблений підхід вже готовий до промислового використання і має потенціал інтеграції у комплексні інтелектуальні транспортні системи, де потрібна висока точність оцінки рівня пального в реальному часі.

Список використаної літератури

1. S. W. Smith, "Moving Average Filters," in *The Scientist & Engineer's Guide to Digital Signal Processing*, 2nd ed., San Diego, CA, USA: California Technical Publishing, 1999. [Online]. Available: dspguide.com/ch15.htm.

2. M. Kirchnerand J. Fridrich, "On Detection of Median Filteringin Digital Images," in IS&T/SPIE Electronic Imaging — Media Forensicsand Security II, Proc. SPIE 7541, 2010

3. А. А. Брус, Е. В. Дикусар, В. С. Ситников, Т. П. Яценко, "Частотный анализ устройства, реализующего алгоритм экспоненциального сглаживания," УСИМ: Упр. системы и машины. – № 1 (237). – 2012. – С. 45–48.
https://jnas.nbuv.gov.ua/j-pdf/USM_2012_1_9.pdf.

4. S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

5. TensorFlow Team. *Keras: The high-level API for TensorFlow*. https://www.tensorflow.org/guide/keras

6. Kingma, D. P., & Ba, J. "Adam: A Method for Stochastic Optimization." *arXiv preprint* arXiv:1412.6980. (Presented at ICLR 2015). arXiv.org

7. FastAPI. *Run a Server Manually — ASGI Servers*. FastAPI Docs. https://fastapi.tiangolo.com/deployment/manually/

8. Scikit-learn Developers. "*mean_absolute_error — scikit-learn documentation*." https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html

9. Tensor Flow Developers. "tf. keras. metrics. Root Mean Squared Error — API documentation." https://www.tensorflow.org/api_docs/python/tf/keras/metrics/RootMeanSquaredError

References

1. S. W. Smith, "Moving Average Filters," in *The Scientist & Engineer's Guide to Digital Signal Processing*, 2nd ed., San Diego, CA, USA: California Technical Publishing, 1999. [Online]. Available: dspguide.com/ch15.htm.

2. M. Kirchnerand J. Fridrich, "On Detection of Median Filteringin Digital Images," in IS&T/SPIE Electronic Imaging — Media Forensicsand Security II, Proc. SPIE 7541, 2010

3. A. A. Brus, E. V. Dykusar, V. S. Sytnykov, T. P. Yatsenko, "Frequency analysis of a device that implements the exponential smoothing algorithm," USYM: Appl. systems and machines. –№ 1 (237). – 2012. – C. 45–48.
https://jnas.nbuv.gov.ua/j-pdf/USM_2012_1_9.pdf.

3. S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

4. TensorFlow Team. *Keras: The high-level API for TensorFlow*. https://www.tensorflow.org/guide/keras

5. Kingma, D. P., & Ba, J. "Adam: A Method for Stochastic Optimization." *arXiv preprint* arXiv:1412.6980. (Presented at ICLR 2015). arXiv.org

6. FastAPI. *Run a Server Manually — ASGI Servers*. FastAPI Docs. https://fastapi.tiangolo.com/deployment/manually/

7. Scikit-learn Developers. "*mean_absolute_error — scikit-learn documentation*." https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html

9. Tensor Flow Developers. "tf. keras. metrics. Root Mean Squared Error — API documentation." https://www.tensorflow.org/api_docs/python/tf/keras/metrics/RootMeanSquaredError

Combined Filtering of Fuel-Level Sensor Data on Mobile Platforms Using Classical Algorithms and an LSTM Network

Y.V. Teneta, V.S. Sytnikov
Odessa Polytechnic National University

Abstract. *The paper addresses noise and inertia-induced distortions in signals from fuel-level sensors on vehicles. A combined approach is proposed that integrates classical smoothing methods (moving average, median filtering) with deep learning based on an LSTM network. To construct reference data, both real and synthetically generated examples were employed. An experimental analysis showed that the combined model outperforms traditional algorithms in terms of accuracy and stability. The results indicate the promise of integrating the approach into transport monitoring systems.*

Keywords: *fuel-level sensors, mobile platforms, signal smoothing, moving average, median filtering, exponential smoothing, recurrent neural network, LSTM, combined filtering, supervised learning, synthetic data.*

Отримано 15.09.2025

Тенета Євген Валентинович, аспірант кафедри комп`ютерних систем, Національний університет «Одеська політехніка»; проспект Шевченка, 1, м. Одеса, 65044, Україна.
E-mail: 9526686@gmail.com

Yevhen Teneta, postgraduate student of the Department of Computer Systems,
Odesa Polytechnic National University; 1, Shevchenko Avenue, Odesa, 65044, Ukraine.
E-mail: 9526686@gmail.com

ORCID ID: https://orcid.org/0009-0003-7917-5689

Ситніков Валерій Степанович, д. т. н., професор, завідувач кафедри комп'ютерних систем, Навчально-науковий інститут штучного інтелекту та робототехніки, Національний університет «Одеська політехніка»; проспект Шевченка, 1, м. Одеса, 65044, Україна. E-mail: sitnikov@op.edu.ua

Valerii Sytnikov, Dr. of Technical Science, Professor, Head of the Department of Computer Systems, Institute of Artificial Intelligence and Robotics, Odesa Polytechnic National University; 1, Shevchenko Avenue, Odesa, 65044, Ukraine. E-mail: sitnikov@op.edu.ua

ORCID ID: https://orcid.org/0000-0003-3229-5096